Universidade de Aveiro
2011

Departamento de Electrónica, Telecomunicações e Informática.

Marco André Almeida Sousa

Receptor de Propagação para Satélite Alphasat

Universidade de Aveiro
2011

Departamento de Electrónica, Telecomunicações e Informática.

Marco André Almeida Sousa

Receptor de Propagação para Satélite Alphasat

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Electrónica e Telecomunicações, realizada sob a orientação científica do Dr. Armando Rocha, Professor auxiliar do Departamento de Electrónica e Telecomunicações da Universidade de Aveiro.

o júri

presidente

Prof. Dr. João Nuno Pimentel da Silva Matos
Professor Associado da Universidade de Aveiro

vogais

Prof. Dr. Victor Daniel Neto dos Santos
Professor Adjunto do Departamento de Engenharia Electrotécnica do Instituto Superior de Engenharia de Coimbra

Prof. Dr. Armando Carlos Domingues da Rocha
Professor Auxiliar da Universidade de Aveiro (Orientador)

Prof. Dr. José Carlos da Silva Neves
Professor Catedrático da Universidade de Aveiro (Co-orientador)

agradecimentos

Em primeiro lugar gostaria de agradecer ao Prof. Armando Rocha pela disponibilidade, paciência e pela ajuda que me deu ao longo do trabalho.
Agradeço ao Paulo Gonçalves do IT pela ajuda na revisão do *hardware* e na soldagem dos componentes mais delicados.
Agradeço à minha família que sempre me acompanhou e apoiou as minhas escolhas.
Por último, mas não menos importante, agradeço aos meus colegas e amigos que dentro e fora do ambiente académico me ajudaram e tornaram mais fácil a conclusão desta etapa.

palavras-chave

Progação, Receptor de *Beacon*, Comunicação por Satélite, *Software Defined Radio*.

resumo

As comunicações por satélite são cada vez mais comuns nos dias de hoje e, com a evolução da tecnologia, mais e mais serviços são desenvolvidos para funcionarem em infra-estruturas deste tipo.

O esgotamento da largura de banda às frequências mais utilizadas conduz à necessidade de exploração de frequências mais elevadas. Os fenómenos de propagação que degradam o sinal agravam-se com o aumento de frequência pelo que os operadores necessitam de modelos para prever o desempenho dos serviços oferecidos. O satélite Alphasat, com data de lançamento para 2012, integra um módulo com *beacons* nas bandas Q e V, o TDP5, para estudo dos fenómenos de propagação nestas bandas.

Neste trabalho descreve-se a construção de um receptor de baixo custo capaz de realizar a medição de amplitude e fase do sinal *copolar* e *crosspolar* da baliza de 39.4 GHz (banda Q) baseado nos conceitos de *Software Defined Radio*. Será dada atenção ao *hardware* do *frontend* analógico que fará a amplificação e conversão de frequência bem como ao *software* que implementará o detector digital para a estimação do sinal.

keywords

Propagation, *Beacon* Receiver, Satellite Comunications, *Software Defined Radio*.

abstract

Satellite communications are widely used nowadays and, with the new technologies being developed, more and more applications are becoming possible.
As the usable frequency bandwidth is getting full it leads to the necessity of exploration of higher frequencies and, with this, measurement campaigns are being made so these frequencies can be considered reliable. The Alphasat satellite by ESA, with launch date scheduled to 2012, is within these campaigns and contains a module with beacons at the bands Q and V, the TDP5, for the study of propagation events at those frequency bands.
In this work we describe the development of a low cost beacon receiver able to measure the *copolar* and *crosspolar* received signal from the 39.4 GHz beacon (Q band) based on *Software Defined Radio* concepts. The work comprises a *down converter* and a flexible digital beacon detector software based on *Software Defined Radio* and a commercial digital radio card.

Índice

Lista de Tabelas

Lista de Figuras

Lista de Acrónimos

ADC	Analog to Digital Converter
CNR	Carrier to Noise Ratio
CO	Copolar
CW	Continuous Wave
CX	Crosspolar
DAC	Digital-to-Analog Converter
DDC	Digital Down Convertion
DDS	Direct Digital Synthesis
DETI	Departamento de Electrónica, Telecomunicações e Informática
DFT	Discrete Fourier Transform
DNL	Differential Nonlinearity Error
DSP	Digital Signal Processor
DUC	Digital Up Conversion
EIRP	Equivalent Isotropic Radiated Power
ENOB	Effective Number of Bits
ESA	European Space Agency
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Array
FTP	File Transfer Protocol
GEO	Geosynchronous Earth Orbit
GPS	Global Positioning System
GUI	Graphical User Interface
HAPS	High Altitude Platform System
I2C	Inter-Integrated Circuit
IF	Intermediate Frequency
IRM	Image Rejection Mixer
LEO	Low Earth Orbit
LNA	Low Noise Amplifier

LO	Local Oscillator
LSB	Least Significant Bit
MMICs	Monolithic Microwave Integrated Circuits
NASA	National Aeronautics and Space Administration
NCO	Numerically-Controlled Oscillator
NF	Noise Figure
NSD	Noise Power Spectral Density
OCXO	Oven-Controlled Crystal Oscillator
OMT	Orthomode Transducer
OX	Cristal Oscillator
PCB	Printed Circuit Board
PGA	Programmable Gain Amplifier
PLL	Phase Locked Loop
RF	Radio Frequency
RMS	Root Mean Square
SDR	Software Defined Radio
SINAD	Signal-to-Noise and Distortion
SNR	Signal to Noise Ratio
SPI	Serial Peripheral Interface
SWIG	Simplified Wrapper and Interface Generator
TCXO	Temperature-Compensated Crystal Oscillator
TDP	Technological Demonstration Payload
UA	Universidade de Aveiro
UHF	Ultra High Frequency
USB	Universal Serial Bus
USRP	Universal Software Radio Peripheral
VCO	Voltage Controlled Oscillator
VHF	Very High Frequency

1 Introdução

O Homem sentiu sempre a necessidade de ultrapassar os seus limites. Começando com a simples invenção da roda, passando pelo importante marco da descoberta da electricidade ou, mais recentemente, a revolução do mundo moderno com a introdução dos computadores pessoais, os progressos conseguidos mostram a determinação da Humanidade em dar um passo mais adiante.

Enquanto para alguns o conhecimento do Espaço era a maior ambição, outros pensavam numa forma de o utilizar para fins que poderiam revolucionar as tecnologias e serviços da altura. Com isto em mente, surgiu a ideia de criar plataformas em órbita com a Terra que fossem capazes de realizar várias funções a partir do Espaço. Tal como os corpos que orbitam à volta dos planetas, estes novos equipamentos foram também apelidados de satélites (embora artificiais) e com eles um novo mundo de possibilidades surgia.

Além da ajuda que os satélites iriam trazer para a conquista do Espaço, destes adviriam outros benefícios que poderiam solucionar alguns problemas da existente infra-estrutura de comunicações. Um dos maiores problemas da altura era sem dúvida a necessidade de comunicação a grandes distâncias. A dificuldade em estabelecer uma ligação rádio entre dois pontos arbitrários da Terra (limitada a frequências relativamente baixas e com pouca capacidade) impedia a exploração de lugares que poderiam contribuir para a expansão da economia de uma região. A aposta na construção de satélites como repetidores de micro-ondas mostrava-se bastante credível. A ideia, que surgiu em 1945, é vista quase como ficção científica. A implementação e exploração só aconteceram vastos anos depois e ganhou uma projecção assinalável no mercado por volta do ano 2000 [1].

Hoje em dia os satélites são imprescindíveis no mundo moderno e asseguram muitos serviços que utilizamos diariamente. Estes permitem fazer desde uma simples chamada de voz até à transmissão de dados como um sinal de televisão, que visualizamos confortavelmente nas nossas casas. A digitalização das comunicações por satélite permitiu uma melhor exploração do espectro contudo novos serviços estão a reclamar bandas superiores à banda K_u (11 a 14.5 GHz) para as quais as condições de propagação na Troposfera são mais adversas.

1.1 Os Satélites

1.1.1 Características básicas

Com a evolução das tecnologias os satélites passaram a ser utilizados para diversas funções. A sua construção é feita desde raiz a pensar na função que irá desempenhar e, consequentemente, cada satélite tem as suas características próprias. Apesar das diferenças em termos de funcionalidade, a estrutura básica é idêntica e segue os mesmos padrões em todos os modelos.

Um satélite é constituído por diversas partes [2]. A nível mecânico é necessário um sistema de medição de posição e controlo de apontamento. Uma parte designada *bus* mantém todos os sistemas do satélite a funcionar de forma correcta, assegurando operações de controlo de altitude, propulsão e gestão de energia e por fim, o *payload* que executa as funções para as quais o satélite foi desenhado. O satélite poderá incluir diversos sistemas como, por exemplo, sofisticados equipamentos de detecção remota ou transponders para a emissão/recepção de sinais de rádio.

A energia é conseguida com base em painéis solares a qual pode ser consumida imediatamente ou armazenada em baterias. Como se sabe, a produção de energia eléctrica utilizando sistemas solares é muito ineficiente. Devido a este detalhe os satélites deverão ser capazes de reposicionar os seus painéis solares de forma a conseguirem o máximo rendimento. A maior parte da energia é transformada em calor. O calor produzido deve ser radiado para o espaço por um eficiente sistema de controlo de temperatura.

Os satélites são mantidos numa determinada órbita. A órbita e posição do satélite dependem da função pretendida para o mesmo. Os satélites geoestacionários (GEO) são bastante comuns em telecomunicações comerciais: a recepção é feita com uma antena fixa e de ganho relativamente elevado. Estes mantêm um movimento sincronizado com a rotação da Terra e encontram-se sobre o plano equatorial numa determinada longitude e a uma distância de cerca de 35800 km.

Apesar dos satélites geoestacionários serem os mais utilizados, outros existem que são essenciais para outro tipo de aplicações. Como exemplo podemos referir os satélites LEO (Low Earth Orbit) que orbitam a Terra várias vezes por dia. Estes satélites são posicionados a distâncias que variam entre os 500 km e os 1500 km e, em comparação com os geoestacionários, têm a vantagem de não ser necessária tanta potência no sinal emitido e o atraso de propagação ser consideravelmente menor devido à menor distância ao satélite. A disponibilidade do sinal numa determinada área é limitada ao tempo em que o satélite se encontra sobre esta.

Infelizmente, tal como qualquer outra peça de tecnologia, os satélites não funcionam para sempre. Chegando ao fim do tempo de vida útil, um novo satélite terá de ser fabricado e posto em órbita para substituir o antigo. Em termos médios, os satélites geoestacionários são desenhados para durarem cerca de 15 anos, um ciclo de vida maior que os não geoestacionários (cerca de 10 anos de duração).

1.1.2 Tipos de Satélite

Como já foi referido anteriormente há diferentes tipos de satélite apesar de apresentarem a mesma estrutura básica. De acordo com a função para a qual foram desenhados estes podem ser divididos em diferentes classes [3]:

- Satélites de Astronomia: Medição de radiação cósmica ou como plataforma de telescópio que beneficia da ausência de distorção devido à atmosfera. O telescópio espacial Hubble é um exemplo deste tipo de satélites;

- Satélites de Comunicações: Sem dúvida os satélites mais utilizados. Estes asseguram serviços como as transmissões de rádio, televisão ou telefone que usamos no nosso dia-a-dia. Os satélite Anik E e Via-Sat são exemplos recentes;
- Satélites de Navegação: Inicialmente criados para conseguir a localização de navios, os satélites de navegação são hoje utilizados correntemente pelos receptores de GPS (Global Positioning System);
- Satélites de Meteorologia: Um dos satélites deste tipo é o Meteosat. Estes satélites utilizam imagens obtidas por vários tipos de sensores (habitualmente passivos) da atmosfera para conseguir informações como temperatura, humidade, detecção de nuvens de cinzas, ocupação do terreno, etc;
- Satélites de Busca e Salvamento: Este tipo de satélites têm como objectivo salvar vidas. Estes são capazes de detectar sinais de socorro de barcos, aviões ou de pessoas com um equipamento emissor em zonas remotas. O satélite Cospas-Sarsat foi desenvolvido para este fim. Actualmente existem vários sistemas dedicados ao *tracking* e *monitoring*. O sistema ARGOS permite, entre outros serviços, seguir a rota de um barco pesqueiro, acompanhar a migração de um grupo de animais ou até proceder à monitorização da temperatura da água do mar ou do nível do caudal de um rio. A informação recolhida pode ser vista em tempo real num website dedicado ou enviada directamente para o utilizador.
- Satélites de Detecção Remota: Estes satélites são utilizados para observar o ambiente que nos rodeia com sensores activos ou passivos. Ocupação dos solos, medição de temperatura de oceanos e crosta, deslocações da crosta terrestre e mapeamento de derrames de petróleo são alguns dos parâmetros medidos. Os satélites de *Remote Sensing*, como o Radarsat e o EnviSat, ajudam-nos a preservar os recursos que são tão importantes para assegurar o nosso futuro.

Os satélites servem de suporte a diversas aplicações. As suas capacidades são deveras importantes no mundo moderno, tanto que cada vez mais serviços estão a ser desenvolvidos para serem utilizados com estes equipamentos.

1.2 Comunicações por Satélite

A necessidade de comunicação tem sido uma preocupação constante. Mais e melhores formas de trocar informação entre locais distantes têm sido criadas ao longo dos anos e hoje em dia temos uma boa infra-estrutura de comunicações que nos permite usufruir dos mais variados serviços. A introdução de satélites no universo das comunicações permitiu levar a informação mais longe superando vários obstáculos que as ligações físicas tinham como limitação. A utilização das fibras ópticas levou a alguma perda de protagonismo do satélite em telecomunicações contudo a ubiquidade do sinal de satélite em zonas remotas e ainda a deficiente oferta de largura de banda em infra-estruturas fixas leva a que seja importante em zonas subdesenvolvidas.

Um satélite de telecomunicações, em termos básicos, funciona como um repetidor de micro-ondas embora os modernos satélites já sejam regenerativos e tenham capacidades

de adaptação ao meio de propagação que enfrentam. O seu *payload* é constituído por receptores, transmissores, uma ou mais antenas e todo o *software* e *hardware* necessário para processamento de informação [4]. Na Figura 1.1 é mostrado um sistema de comunicações por satélite básico entre duas estações terrestres.

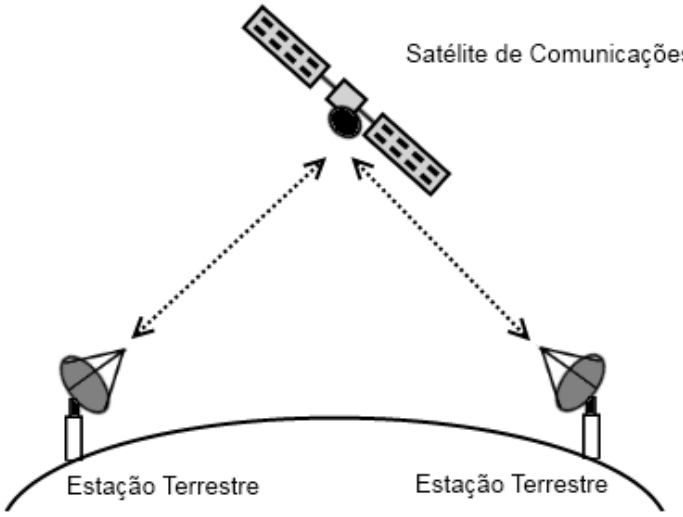


Figura 1.1: Esquema básico de comunicação via satélite.

O satélite de comunicações permite a transferência de informação entre uma ou mais estações terrestres. Como é mostrado na Figura 1.1, uma estação terrestre que pretenda comunicar com outra estação distante radia o sinal para o satélite, o satélite recebe o sinal e retransmite-o para o destino. O sinal transmitido poderá ser recebido por qualquer estação terrestre que esteja dentro da área de cobertura do satélite, o chamado *footprint* do satélite.

Este tipo de sistemas de comunicação é utilizado em diversas circunstâncias. Por exemplo, num sistema de comunicação de micro-ondas terrestre, onde é necessário existir uma linha de vista entre estações ou pontos de replicação do sinal, seriam necessários cerca de 20 repetidores para conseguir uma ligação de 1000 km entre estações de comunicação. Ao recorrer a um satélite é possível conseguir a mesma ligação sem a utilização de qualquer repetidor intermédio, bastando que as estações terrestres tenham o equipamento necessário para a transmissão/recepção por satélite. Tal como outras ligações sem fios, as comunicações por satélite gozam de mobilidade e liberdade independentemente do local, desde que dentro do *footprint* do satélite.

A possibilidade de estabelecer comunicações num curto espaço de tempo poderá ser essencial em casos de catástrofe. Terramotos ou furacões podem danificar a infra-estrutura terrestre de comunicações e, neste caso, os satélites são a alternativa para estabelecer uma ligação rapidamente como aliás ficou patente no terramoto do Haiti.

Uma outra solução para comunicações de emergência e que usa igualmente ligações de micro-ondas são as plataformas de elevada altitude (HAPS). Podem ser lançadas rapidamente e, pelo facto de serem mantidas a algumas dezenas de km de altitude, podem prestar serviços sobre uma grande área. Uma contrapartida relativamente às comunicações terrestres: em condições normais os satélites de comunicações duram em média apenas 15 anos enquanto os cabos dos sistemas terrestres duram uma vida inteira.

Apesar das comunicações por satélite já serem por si só um meio de troca de informação excelente estas podem ainda ser utilizadas em conjunto com as comunicações terrestres. À medida que a evolução das tecnologias avança, mais serviços poderão ser fornecidos via satélite e conjugando os dois tipos de comunicação conseguimos estruturas de transmissão/recepção com uma maior fiabilidade, qualidade e diversidade.

1.3 Propagação em Sistemas de Comunicação via Satélite

O principal problema no uso de micro-ondas em comunicações terrestres ou Terra-Satélite são os fenómenos meteorológicos (nomeadamente precipitação) que acabam por afectar o sinal transmitido.

As bandas Ka (26.5 a 40 GHz) e Q (33 a 50 GHz) disponibilizam uma elevada largura de banda (centenas de MHz) que pode ser utilizada para serviços com elevada taxa de transmissão. A ViaSat é uma das empresas que presta serviços com uma elevada capacidade e disponibiliza um conjunto de ofertas que passam pela criação de redes por satélite para comunicações móveis ou fixas, serviços de banda larga para fins comerciais ou até segurança e encriptação de dados em redes militares. A sua oferta não se resume só ao mercado comercial mas também ao governamental.

O meio de propagação, principalmente a troposfera, pode contudo causar graves contrariedades sendo uma delas a atenuação profunda do sinal que inviabiliza esporadicamente as ligações.

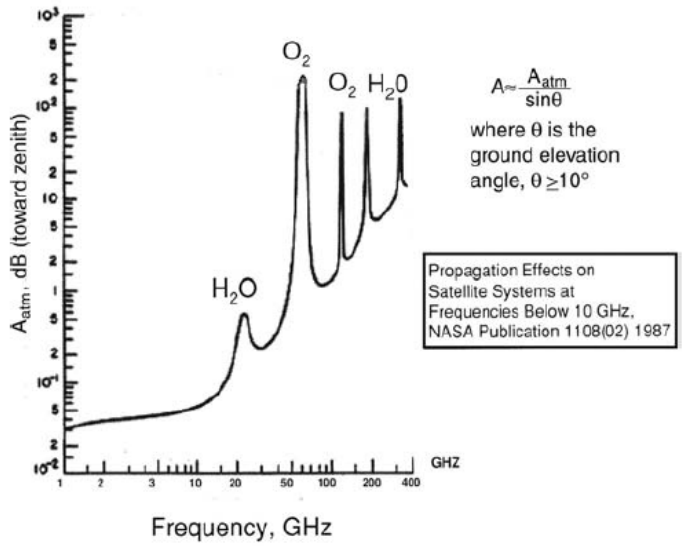


Figura 1.2: Atenuação zenital da atmosfera em função da frequência.

A Figura 1.2 mostra um gráfico da atenuação zenital da atmosfera gasosa (sem precipitação, nuvens e nevoeiro) [1]. A gama de frequências para estabelecer uma ligação com as melhores condições situa-se entre 1 GHz e 4 GHz. As frequências na banda K_u são igualmente utilizadas e os problemas de propagação bastante reduzidos. Acima dos 12

GHz a absorção do sinal começa a ser não desprezável: nos 22.2 GHz existe uma linha de absorção devida ao vapor de água e nos 60 GHz um complexo ressonante que causa uma elevada atenuação.
As frequências dos 20 e 30 GHz (banda Ka) já são usadas para comunicações por satélite [5] [6] [7] pois oferecem uma maior largura de banda e um menor congestionamento que se começa a sentir a frequências mais baixas [8]. A tecnologia é mais dispendiosa, contudo está-se a tornar mais competitiva. A banda Q (em especial os 40 GHz) já suscita ambições para telecomunicações e daí uma motivação para a realização da experiência com o Alphasat.

A constituição da atmosfera e as condições atmosféricas estão, então, relacionadas directamente com a degradação de um sinal de satélite. Em particular, um dos principais factores para a deterioração do sinal em frequências acima dos 4 GHz são os denominados hidrometeoros no estado líquido: gotas de chuva, nuvens e nevoeiro. As gotas de água absorvem a energia (possuem uma constante dieléctrica complexa) e também a espalham causando atenuação. A atenuação aumenta com a frequência.

A implementação de novos serviços a frequências elevadas exige do operador um bom conhecimento do canal de propagação para oferecer ao cliente um serviço especificado com a qualidade que poderá obter. Eventualmente o desenho de sistemas que combatam os efeitos do canal (modulação adaptativa, controlo de potência, etc) requer igualmente um bom conhecimento sobre os aspectos dinâmicos de atenuação do canal: taxa de variação (dB/s) e duração da atenuação.

Assim, campanhas para a medição de diversos fenómenos atmosféricos de propagação são realizadas e, com os resultados obtidos, são desenvolvidos modelos que permitem um dimensionamento dos futuros sistemas de comunicação e, possivelmente, o projecto de sistemas que minimizem as limitações introduzidas pelo canal.

Atenuação, cintilação e despolarização são os fenómenos mais estudados experimentalmente. A atenuação acabou, de certa forma, por ser explicada nos parágrafos anteriores.

A cintilação é consequência da turbulência atmosférica que origina variações do índice de refracção. Daqui resultam perturbações de fase e amplitude na frente de onda que são traduzidas pela antena numa variação de amplitude do sinal em torno do seu valor médio.

A despolarização é provocada essencialmente por gelo e chuva que são partículas não esféricas. Um sinal que sofre despolarização vê parte da potência transmitida numa determinada polarização ser transferida para a polarização que lhe é ortogonal. Nestas condições o sinal que será recebido irá ter duas componentes: o sinal inicialmente transmitido com atenuação, que é chamado de *copolar* (CO) e uma componente ortogonal ao sinal *copolar*, denominado de *crosspolar* (CX). O estudo da despolarização exige a medição das duas polarizações que são separadas na antena receptora por um *Orthomode Transducer*.

O estudo dos fenómenos de propagação tem assim um papel vital na conquista das frequências mais elevadas que estão ainda por explorar.

1.4 Estrutura da Tese

No capítulo 2 irão ser descritas as características de um receptor de propagação, as dificuldades relativas à sua implementação e o estado da arte neste tipo de receptores. Será também apresentado o satélite Alphasat, introduzido o conceito *de Software Defined Radio* e por fim enumerados os objectivos da dissertação.

O capítulo 3 será dedicado ao *hardware* do receptor de propagação. Irá ser feita uma descrição das unidades que constituem o *frontend* analógico e seu funcionamento e ainda apontadas as principais características do kit de rádio que será utilizado na detecção.

Seguidamente, no capítulo 4, serão tratados assuntos relativos ao detector digital. Este capítulo irá explicar alguns conceitos sobre o processamento digital de sinal, descrever os algoritmos utilizados para as estimativas e mencionar as principais ferramentas de *software* usadas.

A implementação do sistema é descrita no capítulo 5. Aqui serão apontadas as principais alterações, melhoramentos e novas funcionalidades tanto no *hardware* como no *software* do receptor. Será ainda detalhado o funcionamento do detector digital.

No capítulo 6 o receptor de propagação irá ser avaliado. Inicialmente serão testadas as várias partes separadamente e de seguida todo o conjunto vai ser montado e testado com sinais realistas.

Por último, o capítulo 7 estará reservado para uma avaliação do trabalho e possíveis melhoramentos e futuras adições.

2 Receptor de Satélite

Neste capítulo contextualiza-se a importância do desenvolvimento de um receptor de propagação abordando-se a sua utilidade na realização de experiências de propagação, a problemática geral no desenvolvimento deste tipo de receptor, as aproximações existentes e as soluções tomadas.

2.1 Características genéricas de um receptor

O desenho de um receptor depende em parte das características do sinal como, por exemplo, o tipo de modulação ou a frequência da portadora.

Um sinal genérico de satélite é semelhante ao de qualquer outro sistema de comunicações: uma portadora com modulação (de fase habitualmente ou fase e amplitude). A portadora é a frequência a que o sinal é enviado enquanto a modulação diz respeito à informação que se pretende transmitir. Existem vários tipos de modulação disponíveis. Estas podem ser analógicas ou digitais, no entanto, hoje em dia cada vez mais se opta pelas digitais.

Para receber o sinal do satélite é necessária uma antena correctamente dimensionada, uma unidade de acondicionamento de sinal e ainda um desmodulador. O ganho da antena deverá garantir uma relação sinal ruído suficiente. Como, por limitação de *hardware*, não podemos ainda desmodular o sinal à frequência da portadora, é necessária a conversão da frequência do sinal para sucessivas frequências mais baixas.

2.1.1 Dificuldades de Implementação

Ao projectar o receptor para o sinal de satélite é necessário ter em conta alguns factores que irão afectar o seu desempenho. Estes factores são comuns ao projecto de qualquer receptor de rádio. Destes destacam-se:

- O ruído de amplitude. O ruído - ruído atmosférico ou o produzido pelos componentes é incontornável. O ruído gerado pelos componentes deve ser minimizado;
- Linearidade dos componentes utilizados. Como exemplo, um amplificador de RF não funciona da mesma forma para todas as frequências, temperaturas ou potências. É necessário ter em atenção as zonas normais de funcionamento dos componentes e as suas limitações;
- Interferências e rejeição de frequência imagem. As interferências devem ser evitadas e a frequência imagem deve ser filtrada antes da conversão de frequência sob pena de, no mínimo, piorar a SNR (relação sinal-ruído) em 3 dB;

- Ruído de fase. Os osciladores devem possuir baixo ruído de fase pois este é transferido ao sinal espalhando a sua potência na frequência o que dificulta a sua posterior sincronização.

Em resumo: o receptor deverá ser capaz de amplificar o sinal RF, convertê-lo para uma interferência intermédia (IF) ou para uma banda base e proceder à sua detecção tentando sempre minimizar os problemas mencionados.

2.2 Receptor de Propagação

O receptor que aqui se pretende desenvolver tem contudo requisitos ligeiramente diferentes: pretende-se medir, com a máxima gama dinâmica, a amplitude de um (*copolar*) ou dois sinais (*copolar* e *crosspolar*) não modulados (ou seja CW) de um *beacon*. Estes sinais "transportam" o testemunho dos efeitos sofridos pela propagação.

O satélite pode possuir ainda facilidades direccionadas para o teste de sistemas de comunicação ou experiências de propagação. O módulo para experiências de propagação denomina-se de *propagation payload* que radia um sinal CW denominado *beacon* (padrão) de satélite.

Um *beacon* de satélite é um sinal de baixa potência que pode ser utilizado para funções de controlo, telemetria e pesquisa. Os *beacons* normalmente são não modulados e são transmitidos com uma potência constante. Ora, como se pretende efectuar uma análise contínua do sinal recebido, as características destes sinais são excelentes para experiências de propagação.

Estes receptores deverão fazer o seguimento do sinal recebido e ser capazes de o recuperar em situações de atenuação profunda causado pelas condições atmosféricas adversas. A análise das variações e das características do sinal recebido ao longo do tempo irá posteriormente dar-nos a informação necessária para o estudo dos fenómenos que afectam a propagação.

A construção de um conjunto receptor para o *beacon* de satélite segue a mesma topologia que qualquer outro receptor de rádio. É necessária uma antena correctamente dimensionada, um *frontend* para acondicionamento de sinal e um detector de elevada sensibilidade.

O receptor de propagação que se pretende implementar neste trabalho irá medir então o *beacon* na banda Q (*copolar* e *crosspolar*) a partir de uma 1ª IF a cerca de 2 GHz. O *beacon* será transmitido numa frequência próxima dos 40 GHz a partir do futuro satélite AlphaSat.

2.3 Alphasat

Desenvolvido pela ESA (European Space Agency) [9], o satélite Alphasat tem data de lançamento prevista para o ano 2012 a bordo do foguetão espacial Ariane 5.

O Alphasat, que será colocado numa órbita quasi-geoestacionária, irá desempenhar funções de comunicação avançadas, aumentando a área de cobertura da actual rede de

satélites da Inmarsat [10]. O *footprint* irá estar centrado em África, com uma área de cobertura que chegará à Europa, Médio Oriente e a algumas partes da Ásia. O satélite será previsivelmente operado com uma inclinação máxima de cerca de 3^{o} o que dificultará um pouco o seguimento e irá introduzir Doppler nos sinais.

Além dos serviços suportados pelo seu *payload*, o satélite possui ainda um conjunto de módulos para fins de estudo e desenvolvimento. Estes módulos são chamados de TDP's (Technical Demonstration Payloads) e o satélite incorporará quatro dispositivos deste tipo.

O sinal que pretendemos medir provém do TDP5. Este módulo radiará dois *beacons*: um a 19.701 GHz e outro a exactamente o dobro da frequência ou seja 39.4 GHz com vista à realização de experiências de propagação.

Na Figura 2.1 está mostrada a estrutura do módulo TDP5 do Alphasat.

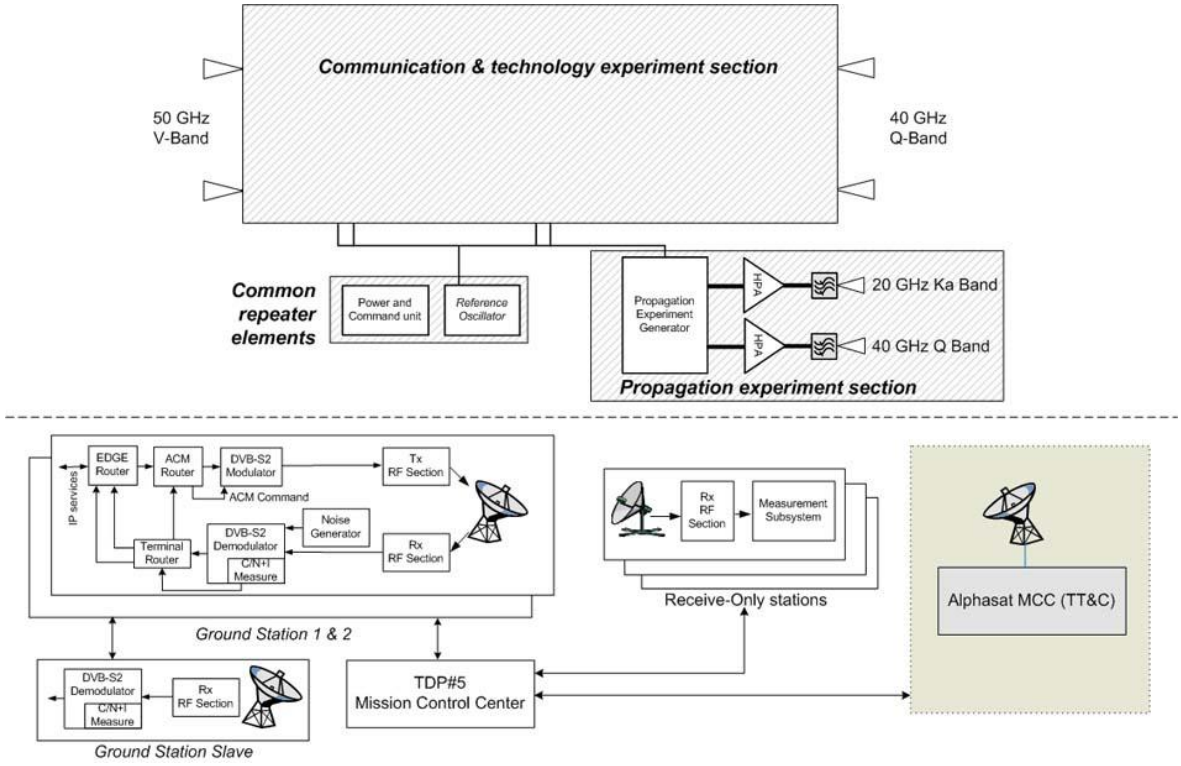


Figura 2.1: Características do TDP5 do satélite Alphasat [11].

Embora ainda não se saibam todos os dados relativos ao sinal radiado do TDP5, as informações já disponíveis são suficientes para o projecto do receptor. Espera-se que o sinal tenha as seguintes características:

- Frequência: 39.402GHz;
- EIRP (Equivalent Isotropic Radiated Power): 26.5 dBW;
- Largura Espectral: 50 a 100 Hz;
- Desvio de Frequência por Doppler: No máximo 3 kHz/dia;
- Variação de potência: 0.5 dB/dia, em condições normais.

O efeito de Doppler deve ser tido em conta na construção da parte de acondicionamento do sinal.

Novas tecnologias têm surgido para a criação de sistemas de rádio na sequência de importantes desenvolvimentos na electrónica analógica e digital. Os primeiros receptores de rádio eram totalmente analógicos. Os processos de conversão e filtragem, bem com a detecção eram conseguidos graças à utilização de componentes electrónicos, o que levava à construção de enormes cadeias de *hardware*. Actualmente, a introdução da tecnologia digital neste tipo de circuitos levou a um novo conceito que está a revolucionar o universo das telecomunicações: o *Software Defined Radio* (SDR) [12].

Em termos sucintos, um SDR é um rádio em que o processamento do sinal é realizado na íntegra por *software*, ou seja, componentes como amplificadores, filtros, mixers ou mesmo detectores são substituídos por equivalentes em *software*. Apesar do conceito remeter para a criação de sistemas de rádio totalmente por *software*, a tecnologia actual não permite o processamento de sinal para frequências elevadas. Este facto implica que seja sempre necessária a utilização de processamento analógico para a conversão de sinais para IF's mais baixas ou para a banda base.

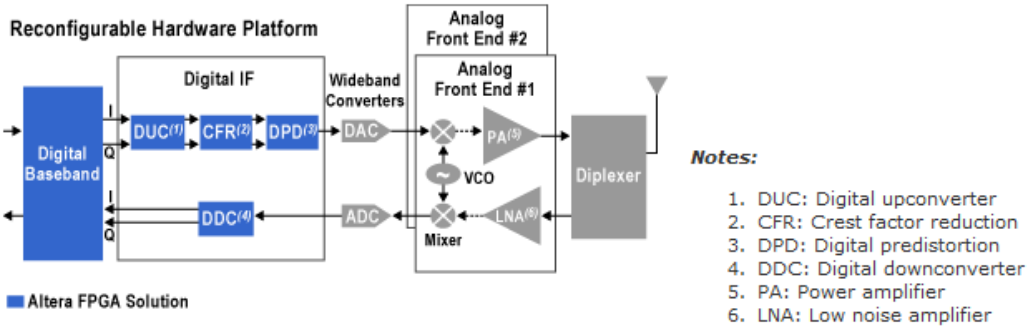


Figura 2.2: Esquema de um SDR actual para comunicações 3G.

Na Figura 2.2 pode ver-se a estrutura de um sistema SDR que pode ser implementado com a tecnologia já existente. Este esquema é baseado numa FPGA (Field Programmable Gate Arrays) da Altera [13] e tem como função a recepção/transmissão de um sinal 3G utilizado em comunicações móveis.

Com a introdução das FPGA's nas comunicações RF, o conceito de *Software Defined Radio* tem vindo a tornar-se cada vez mais uma realidade. Uma FPGA funciona como um processador reconfigurável. Esta está desenhada para efectuar cálculos pesados, mas de forma repetitiva e pode ser reprogramada por *software* para realizar um sem número de operações diferentes. A possibilidade de fácil reconfiguração do *hardware* permite efectuar alterações de forma rápida quando um novo serviço é integrado, sem necessidade de modificar o *hardware*.

O receptor, de acordo com o modelo apresentado, irá ser constituído por um *frontend*, um conversor analógico para digital (ADC) e uma unidade que fará o processamento digital: neste caso uma FPGA. O *frontend* irá filtrar, amplificar e converter o sinal para uma frequência mais baixa. O ADC tratará de converter o sinal para o domínio digital para de seguida entrar na FPGA. A FPGA poderá converter o sinal para a banda base, caso seja

necessário, e procederá à filtragem e decimação do sinal. Este sinal irá, posteriormente, ser introduzido num computador que fará a análise e detecção do mesmo.

Os sistemas SDR apresentam diversas vantagens: permitem diminuir os custos de implementação; reduzem o tempo necessário para o desenvolvimento e facilitam a resolução de problemas em fases de teste.

O facto do processamento digital do sinal ser efectuado por *software* através de funções matemáticas permite que o receptor possa adaptar-se a novas condições que possam surgir em tempo real, o que não acontecia com os receptores puramente analógicos. Para além do que já foi dito, os sistemas SDR têm ainda a vantagem de ter um tempo de vida maior do que os sistemas analógicos. Se não reparemos: o sinal tem como destino final um computador que fará a sua análise e detecção. Os computadores têm a sua capacidade de processamento limitada pelo número de cálculos que o seu processador consegue realizar. À medida que a tecnologia avança melhores processadores surgem a preços que baixam rapidamente. Com eles mais operações poderão ser realizadas e mais complexo e funcional poderá ser o nosso código.

A principal limitação da utilização do SDR está na necessidade de conversão para frequências mais baixas antes do processamento digital. Infelizmente, nos sistemas SDR actuais será sempre necessária a implementação de um *frontend*, em grande parte devido à velocidade dos ADC's que hoje em dia estão disponíveis no mercado.

O receptor que será implementado neste projecto será baseado no *Software Defined Radio* e fará uso de um kit rádio digital apelidado de USRP (Universal Software Radio Peripheral) e de um *software* compatível com esta plataforma: o GNU Radio. Estes temas serão abordados nos próximos capítulos.

2.4 Estado da Arte em Receptores de Propagação

As experiências sobre propagação de sinais de *beacon* de satélite são bastante comuns e existem diversas iniciativas em vários pontos do planeta. Os receptores desenvolvidos para medições dos fenómenos de propagação dependem das características do sinal e cada um é construído de raíz não havendo no mercado um produto comercial para este efeito.

O primeiro receptor desenvolvido na UA, para o satélite Olympus, era completamente analógico. O detector síncrono usado envolvia uma malha de seguimento de fase que é bastante difícil de administrar: pode perder o sincronismo com o sinal atenuado, tem que ser auxiliada na aquisição e implica uma série de ajustes morosos e problemáticos.

Actualmente, no Departamento de Electrónica, Telecomunicações e Informática (DETI) está activa uma experiência de propagação com satélite HotBird-6. O receptor utilizado nesta recolha de dados é em grande parte analógico. O acondicionamento do sinal é feito em cadeias de *hardware* até uma frequência áudio à qual o sinal é adquirido a cerca de 400 kS/s e detectado por técnicas de análise espectral.

A implementação de receptores deste tipo poderá ser simplificada pelo processamento digital nas comunicações RF. O processamento digital do sinal leva a que seja necessário menos tempo de desenvolvimento e teste, sejam reduzidos os custos, resultando ainda em conjuntos bem mais compactos que os sistemas totalmente

analógicos. Foram desenvolvidos protótipos [14] de detectores digitalizando uma IF a 10.7 MHz usando DDC (*Digital Down Converters*) e ADC. Os dados digitalizados eram enviados para um kit DSP o qual fazia a detecção síncrona e uma PLL por *software* que usava o NCO (*Numerically Controlled Oscillator*) da DDC como VCO. O sistema era bastante versátil: o *software* vigiava a CNR do sinal sincronizado e administrava a malha PLL de forma à re-aquisição se efectuar quase de imediato após severa atenuação.

Apesar do sucesso e das inspiradoras ideias o problema era ainda a complexidade do sistema: *hardware* desenvolvido, fontes de alimentação e kit DSP autónomo e uma notável necessidade de conhecimentos muitos vastos e transversais para o implementar. Com a chegada dos sistemas SDR e sua evolução ao longo do tempo novas possibilidades surgiram na construção de sistemas de rádio digitais e soluções de mercado bastante compactas com possibilidades adicionais que são bastante baratas se atendermos aos custos do desenvolvimento. Várias entidades começaram a desenvolver plataformas de processamento digital que podem ser reconfiguradas facilmente e assim criar receptores. Um exemplo é o já referido USRP, desenvolvido pela Ettus Research que, juntamente com o GNU Radio, irão ser a base do receptor que pretendemos implementar.

Uma implementação do detector com base neste kit passa desde logo pela aproximação baseada em análise espectral mas, como veremos adiante, a implementação de técnicas com uma maior gama dinâmica usando por exemplo malhas de seguimento de frequência em tempo real são igualmente possíveis.

Um receptor de propagação deste tipo foi desenvolvido em conjunto em duas teses de mestrado [15] [16]. Ao ser necessária a implementação de um *frontend* analógico antes do tratamento digital, a construção do receptor foi dividida em duas partes. A primeira parte é relativa a um *frontend*, capaz de amplificar e converter o sinal para uma frequência mais baixa e uma outra relativa ao processamento de sinal onde é feita a análise e detecção, sendo esta última ainda capaz de actuar sobre o *frontend* para reconfigurar o oscilador local. Apesar de tanto a parte relativa ao *frontend* como a parte de processamento digital do sinal apresentarem resultados satisfatórios, o conjunto nunca chegou a ser testado na sua totalidade de forma conveniente.

Quanto à oferta de mercado em receptores, esta é muito reduzida e muito dispendiosa. O orçamento para a criação de um único receptor com as especificações que pretendemos acaba por ter custos muito elevados devido a não haver produção em série.

2.5 Objectivos

A presente dissertação vem continuar o trabalho já realizado em [15] e [16], sobre a criação de um conjunto receptor *software* mais *hardware* capaz de efectuar medições de propagação sobre um sinal de frequência 40 GHz.

Os principais objectivos são:

- Optimização do hardware do receptor, uma vez que são conhecidos mais detalhes sobre as características do *beacon* do satélite;
- Desenvolvimento de módulos para fontes de ruído que serão utilizados em testes ao sistema;

- Resolução de problemas de *software* que inviabilizavam a aquisição continuada;
- Revisão dos algoritmos para a estimativa das características do sinal recebido e respectiva validação (frequência, CNR, NSD, etc);
- Redesenho da interface gráfica e introdução de funcionalidades que permitam visualização de dados de passado recente;
- Implementação de mecanismos para realização de cópias de segurança dos ficheiros adquiridos;
- Armazenamento directo em formato Matlab;
- Inclusão de mecanismos automáticos de detecção e alerta perante anomalias ou excepcionalidades nos dados;
- Avaliação de todo o sistema com sinais realistas.

O trabalho estará dividido em duas partes principais: o *hardware* e o *software* do receptor.

3 O Hardware

Neste capítulo irá ser feita uma descrição do *hardware* bem como a explicação dos componentes que serão utilizados. Uma vez que o *hardware* do receptor não sofre grandes alterações relativamente à versão original desenvolvida [16], a descrição respeitante ao *hardware* a construir irá ser breve e resumir-se-á ao que será implementado e aos problemas associados. As novas adições relativas ao *hardware* do receptor irão ser discutidas no capítulo 5, relativo à construção global do sistema.

O sinal proveniente do *beacon* de satélite irá ser transmitido numa frequência de 39.4 GHz com uma potência relativamente baixa. Para ser possível a detecção do sinal, pretende-se converter a frequência original para uma frequência mais baixa de forma a ser possível uma digitalização do sinal pelo ADC. A redução da frequência original não irá ser feita num passo apenas, mas sim através de duas conversões intermédias. A primeira IF será conseguida com um *down converter* comercial e centrar-se-á nos 2 GHz e a segunda IF, de 10.7 MHz, irá ser obtida com o *hardware* que se pretende implementar.

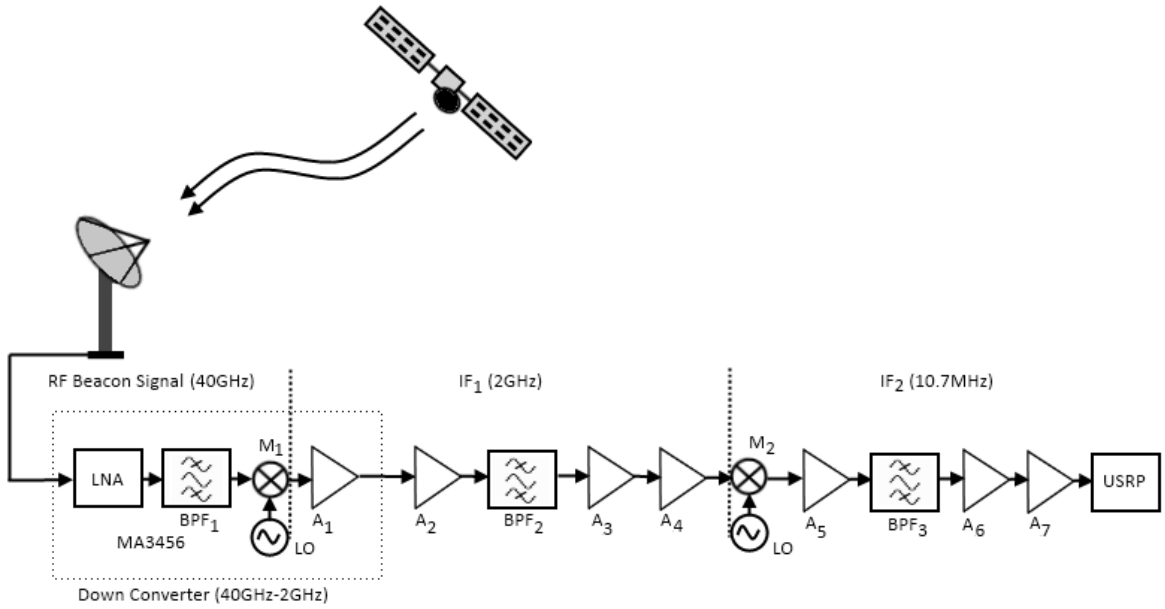


Figura 3.1: Esquemático de *hardware* do Receptor de Propagação.

Na Figura 3.1 está mostrado um diagrama de blocos do *hardware* que será montado. Todo este capítulo irá basear-se neste esquemático.

3.1 Unidade de Acondicionamento e Conversão do Sinal

Estes módulos de *hardware* disponibilizam um sinal com uma amplitude conveniente para a digitalização no kit rádio digital. O cálculo do *Link Budget* permite averiguar qual o ganho total necessário e a relação CNR que teremos à entrada do sistema de aquisição.

Normalmente o experimentador impõe uma gama dinâmica para a medição da atenuação que considere suficiente para os seus objectivos de investigação. Não se esperam sistemas comerciais com uma elevada margem de atenuação (eventualmente 8 a 12 dB). Assim, embora cientificamente fosse interessante realizar medidas com uma maior gama, um objectivo realista para um experimentador sem grandes infra-estruturas instaladas será cerca de 25 a 30 dB. Uma antena para a frequência de 39.4 GHz é bastante dispendiosa devido à enorme precisão requerida da superfície reflectora e do OMT. Por outro lado uma antena muito directiva requer uma elevada precisão de apontamento e consequentemente um sistema de posicionamento bastante caro.

3.1.1 *Link Budget* do Sistema

Para a implementação do receptor é necessário saber as características do sinal à entrada do sistema, bem como os condicionamentos introduzidos pelo próprio receptor no sinal. Desta forma o *link budget* será realizado para garantir à saída do sistema um *signal to noise ratio* (SNR) e gama dinâmica aceitáveis para uma detecção correcta.

Em primeiro lugar será importante saber qual a potência do sinal recebido pela antena. É possível conseguir este valor com a fórmula de Friis (3.1), que relaciona a potência recebida com a frequência do sinal transmitido, o ganho da antena receptora, as perdas na propagação e a potência do sinal radiado pela antena do transmissor.

$$Pr = EIRP.\, G_t.\, G_r \left(\frac{\lambda}{4\pi.\, R}\right)^2 \tag{3.1}$$

Considerando um ganho para a antena receptora G_r de 45dB, um EIRP do *beacon* de 26.5 dBW e uma distância *R*, entre as antenas do emissor e do receptor, de 38453 *km* chega-se a um valor para a potência recebida P_r de -144.55 dBW.

Um outro ponto importante será a determinação da densidade espectral do ruído *N* (dBW/Hz). À entrada do sistema *N* é dado pela expressão (3.2).

$$N = 10 \log\left(K.\left(T_{eq} + T_{Sky}\right)\right) \tag{3.2}$$

Onde T_{eq} é a temperatura de ruído do receptor e T_{Sky} a temperatura da atmosfera que terá um pior caso de cerca de 290 K. O factor de ruído não é mais que a degradação da SNR da entrada para a saída de um bloco do sistema. Iremos então começar por determinar o factor de ruído total F_{nT} para uma cascata de *i* blocos usando a fórmula de Friis para o ruído.

$$F_{nT} = F_{n1} + \frac{F_{n2} - 1}{G_1} + \frac{F_{n3} - 1}{G_1 G_2} + \cdots + \frac{F_{ni} - 1}{G_1 G_2 \ldots G_{i-1}} \tag{3.3}$$

O factor de ruído total do receptor irá depender dos blocos constituintes do receptor (ganho e factor de ruído de cada um) onde cada bloco dá a sua contribuição para o factor de ruído global. Como se pode verificar na expressão (3.3), os primeiros blocos são aqueles que mais contribuem para o factor de ruído. Ao longo da cascata o ganho é cada

vez maior, o que torna as últimas parcelas praticamente insignificantes no cálculo do factor de ruído total do receptor.

No receptor que se pretende implementar, os primeiros blocos a considerar serão um OMT que fará a separação entre o sinal CO e CX, e um *down converter* MA3456. Uma vez que o *down converter* comercial apresenta um ganho consideravelmente elevado (35 dB) os blocos seguintes poderão ser desprezados para o cálculo do factor de ruído total. Sabendo que, de acordo com os fabricantes, o F_n do OMT é 1.25 (F_{n1}) e o F_n do *down converter* é 3.98 (F_{n2}) obtém-se um factor de ruído total do sistema F_{nT} de aproximadamente 5.

O factor de ruído total F_{nT} encontra-se relacionado com a temperatura equivalente do ruído T_{eq} da seguinte forma:

$$T_{eq} = T(F_{nT} - 1) \tag{3.4}$$

onde *T*, é a temperatura de ruído de referência (290 *K*). A temperatura equivalente do ruído é 1160 *K*.

Finalmente, com recurso à expressão (3.2) e sabendo que k corresponde à constante de Boltzman obtém-se uma densidade espectral de ruído de -197 dBW/Hz.

Com os valores conseguidos é possível agora determinar um valor estimado para a CNR do sistema. Tendo em conta que o sinal disponível à entrada do receptor terá -144.55 dBW e que a densidade espectral de ruído será de -197 dBW chega-se a um valor estimado para a CNR de 52 dB. Este valor será o esperado nas melhores condições de propagação possíveis, ou seja, no caso de céu limpo.

Se o seguimento do sinal for efectuado por uma PLL é previsível que a largura de banda de ruído do sistema (LB) seja de cerca de 100 Hz. Assim usando a relação:

$$CNR = SNR + 10\log(LB) \tag{3.5}$$

com SNR mínimo de 10 dB para um sincronismo fiável, podemos estimar uma gama dinâmica de 22 dB. Caso se opte por uma detecção espectral o *offset* introduzido no *copolar* para a máxima atenuação será cerca de 0.4 dB que é um valor aceitável. Estes valores correspondem à utilização de uma antena com cerca de 0.8 m de diâmetro que é conveniente para operar em interiores como se pretende neste caso.

3.1.2 Ganho do Sistema

Sabendo agora a potência do sinal do *beacon* de satélite à entrada do receptor e as características a nível do ruído, é necessário agora dimensionar o ganho do sistema para cumprir os requisitos à entrada do USRP.

O USRP deverá ter na sua entrada um sinal com uma amplitude de 1 V_{pp}, o que corresponde a -20 dBW. Com um sinal de -145.55 dBW no início da cadeia de *hardware*,

já com a adição de 1 dB de perdas devido ao OMT, é necessário um ganho de 125 dB para cumprir os requisitos. Este valor é algo elevado, o que pode causar alguns problemas na construção do *hardware*. Não pode ser, de todo, descartada a hipótese da possibilidade de oscilação, principalmente no caso da IF mais elevada de 2 GHz se for concentrado neste andar todo o ganho. Além disso, é necessário ainda ter cuidado com a linearidade dos amplificadores, o que remete para o conhecimento das suas características como o IP3 ou ponto de compressão de 1 dB. O ponto de compressão de 1 dB refere-se à potência máxima do sinal que o amplificador pode ter à entrada quando na saída se obtém uma potência 1dB abaixo do esperado. O IP3, ou ponto de intersecção de terceira ordem, é o ponto para o qual a curva de intermodulação intercepta a curva linear da potência de saída. Visto isto, a linearidade dos amplificadores poderá ser posta em causa se estes não estiverem a funcionar nas suas zonas normais de trabalho.

Olhando de novo para a Figura 3.1, o ganho que se pretende conseguir será ser repartido pelo LNA e pelas duas IF's do sistema. Após a primeira conversão de IF proporcionada pelo *down converter* comercial e a adição de 35 dB de ganho que este componente proporciona, três amplificadores preparados para funcionar em frequências que englobam os 2 GHz e outros três amplificadores dimensionados para os 10.7 MHz fornecem os restantes 90 dB necessários. Esta divisão visa prevenir possíveis oscilações no sistema que afectariam o seu desempenho. A escolha dos amplificadores deverá ser feita de uma forma cuidadosa, principalmente no caso em que se lida com frequências relativamente elevadas. Os amplificadores a utilizar são circuitos integrados MMIC's da Mini-Circuits [17] pelas boas características que apresentam.

De notar ainda que para se obter o ganho esperado, o dimensionamento deverá ter sempre em conta as perdas que acontecem ao longo da cadeia. As perdas relacionadas com os filtros, divisão e junção de sinais e conversão de frequência deverão ser contabilizadas.

Uma análise mais detalhada encontra-se em [16].

3.1.3 Conversão de IF

Ao longo da cadeia de *hardware*, pretende-se que a frequência do sinal seja reduzida até a um ponto aceitável onde se possa passar confortavelmente para o domínio digital. A frequência original do sinal (39.4 GHz) irá sofrer duas reduções para IF's mais baixas. A primeira conversão será feita pelo *down converter* comercial (IF_1 de 2 GHz) e a segunda com o *hardware* que se pretende implementar (IF_2 de 10.7 MHz).

O princípio para a conversão é bastante simples: dois sinais à entrada do mixer, em que um é o sinal RF e o outro é a referência (LO), originam um sinal à saída com uma frequência que corresponde à diferença das frequências dos dois primeiros sinais e a soma que normalmente é filtrada.

Consultando o esquemático global do hardware do receptor (Figura 3.1), temos a representação de dois mixers para a conversão de IF. O primeiro mixer (M_1) faz parte do módulo comercial a adquirir. Nesta parte não será feito qualquer dimensionamento e então iremos apenas considerar que o sinal sofrerá a redução para a IF_1 de 2 GHz e que terá um ganho adicional dado pelo referido módulo. O segundo mixer (M_2) é que irá ter

toda a atenção. Este bloco irá tratar de converter a IF_1 para a IF_2 e será construído juntamente com o restante *hardware* do receptor a implementar.

A conversão para a IF_2 é realizada às custas de um misturador de rejeição de imagem (IRM). A frequência imagem é uma frequência cuja potência, no processo de conversão de frequência, é convertida para a mesma frequência intermédia do sinal.

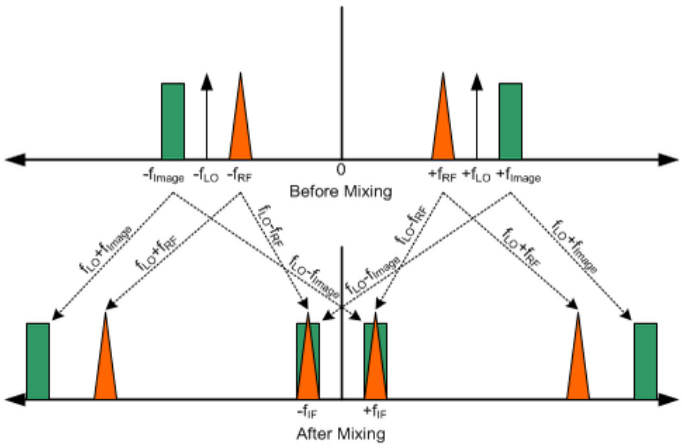


Figura 3.2: Problema da frequência imagem na *down convertion* [18].

Após a conversão o ruído na frequência imagem soma-se ao ruído do sinal degradando a SNR em 3 dB. A supressão da frequência imagem pode fazer-se com filtros se a frequência imagem estiver razoavelmente afastada do sinal (digamos pelo menos 5 a 10% do oscilador local) ou com mixers adequados que a cancelem. Assim, a solução implementada passa pela utilização de um mixer de rejeição de imagem numa arquitectura Hartley já que se pretende uma conversão para uma IF bastante baixa num só estágio, o que reduz a complexidade do *hardware* (poupa uma IF e um oscilador local).

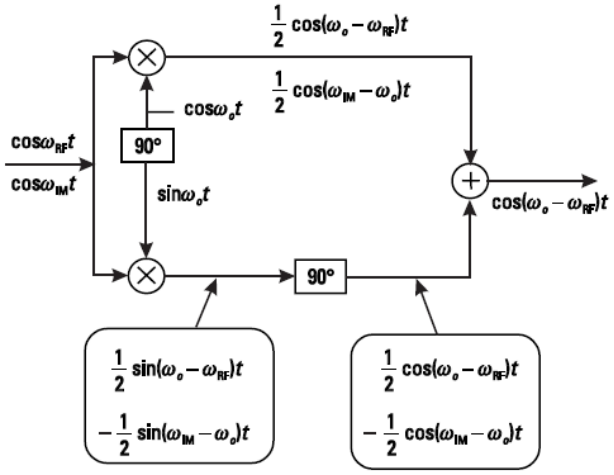


Figura 3.3: Arquitectura Hartley para mixer de rejeição de imagem [19].

O modelo apresentado irá cancelar a frequência imagem através da soma da frequência imagem com uma cópia desta em oposição de fase.

Em termos práticos, o sinal RF e o sinal do LO (oscilador local) irão ser divididos em duas cópias por dois *power splitters* independentes. Uma das cópias do LO será desfasada de 90°, com a introdução de uma linha de transmissão de comprimento $\lambda/4$, e irá entrar

num dos mixers juntamente com uma das cópias do sinal RF. O segundo mixer irá ter nas suas entradas o outro sinal RF proveniente do *power splitter* e a segunda cópia do oscilador local sem qualquer tipo de desfasamento. Por fim, os sinais resultantes dos dois *mixers* irão juntar-se num *power combiner*. Este novo *combiner* tem a particularidade de provocar um desfasamento de 90° numa das portas, desfasamento este que perfaz os 180° necessários para haver o cancelamento pretendido. À saída do *power combiner* teremos então apenas o sinal RF que pretendemos convertido para a nova frequência intermédia IF_2.

Para que a componente da frequência imagem seja eliminada o desfasamento de 180° terá que ser preciso e a amplitude das frequências imagem provenientes dos dois mixers deverá ser a mesma. Se isto não acontecer, não iremos conseguir um cancelamento perfeito e parte do ruído da frequência imagem irá sobrepor-se ao ruído do sinal RF que pretendemos receber.

A escolha dos *power splitter/combiners* e dos mixers utilizados depende, entre outros aspectos, das frequências de trabalho. Os *splitters* (SNC-2-22 da Minicircuits) para a divisão inicial do sinal RF e do LO, tal como os mixers (ADE-R20+ da Minicircuits), estão dimensionados pelo fabricante para funcionar numa gama que engloba os 2 GHz. O *combiner* (JYPQ-16 da Minicircuits) está desenhado para funcionar nos 10.7 MHz.

Apesar de, com este processo, ser possível a eliminação da frequência imagem, isto não dispensa a utilização de filtros antes da conversão. A filtragem posterior poderá eliminar espúrias e reduzirá a potência de ruído fora da banda útil. A descrição dos filtros utilizados irá ser feita no capítulo 5, uma vez que os filtros das duas IF's foram modificados para cumprir os requisitos do sinal de satélite segundo as novas informações disponíveis.

As perdas nos filtros, as perdas de conversão nos mixers e as perdas nos *power splitter/combiners* devem ser levadas em conta no dimensionamento do ganho total do sistema (como foi referido no subcapítulo relativo ao ganho de sistema).

3.2 Unidade de Síntese

No subcapítulo anterior foi introduzido o conceito de conversão de frequências para IF's mais baixas tirando partido das características dos mixers. Foi dito que através da diferença da frequência do sinal RF e do oscilador local (LO) iríamos conseguir uma IF mais baixa que podia ser introduzida nos ADC's. Resta-nos agora perceber o funcionamento do LO bem como as características da sua construção.

Um oscilador é um circuito que produz um sinal periódico que pode ser utilizado em diversas aplicações. Recorrendo ao esquemático da Figura 3.1, o oscilador local será o LO de ataque ao mixer M_2, o qual fará a conversão da IF_1 de 2 GHz para a IF_2 de 10.7 MHz.

O sinal recebido pelo satélite irá variar alguns kHz ao longo do dia. Após a conversão para a segunda IF, um filtro com uma largura de banda estreita de ±7.5 kHz é aplicado ao sinal. Ao existir uma variação de frequência do sinal superior à largura do filtro, o sinal irá ser atenuado por cair fora da largura de banda do filtro. O oscilador local a projectar deverá ter uma frequência programável de forma a acomodar as variações sistemáticas de frequência do satélite e osciladores locais do receptor.

Existem várias arquitecturas para a implementação de osciladores locais. No nosso caso em particular, o LO será realizado com um sintetizador de frequências baseado numa PLL (Phase Locked Loop). Em termos sucintos: um sinal de baixa frequência de um oscilador a cristal com excelente ruído de fase, que servirá como referência, irá ser multiplicado por um determinado factor. Esta explicação mostra o princípio da síntese de frequências, ou seja, partindo de uma frequência de referência é possível gerar novas frequências múltiplas da frequência original.

O principal cuidado a ter na construção de osciladores de referência para conversão de frequências está no ruído de fase.

a) Oscilador ideal. b) Oscilador com ruído de fase.

Figura 3.4: Ruído de fase em osciladores.

O ruído de fase pode ser visto como variações aleatórias na fase de um oscilador que se sobrepõem ao incremento linear com o tempo. Se o ruído de fase do oscilador local for elevado dá origem ao espalhamento da potência do sinal no espectro reduzindo a gama dinâmica das medidas. Em termos ideais, um oscilador deveria ter na sua saída um sinal com o aspecto da Figura 3.4 a) visto no domínio das frequências, no entanto, em situações reais isto não acontece. O sinal proveniente do oscilador será como o mostrado na Figura 3.4 b. Como este problema não pode ser totalmente eliminado o oscilador a projectar deverá ter a melhor estabilidade de fase possível, sendo este um factor de mérito dos osciladores. Uma abordagem interessante ao espectro do sinal de um oscilador livre pode encontrar-se em [20].

Para melhor perceber os conceitos referidos e o funcionamento global do LO, os pontos seguintes irão fazer uma breve descrição dos principais constituintes do oscilador local.

3.2.1 PLL

Uma PLL funciona sobre um princípio que visa a comparação das fases de dois sinais. A diferença entre as fases irá ser utilizada para controlar a frequência de um oscilador controlado por tensão (VCO) [21].

Uma melhor compreensão poderá ser conseguida analisando os blocos constituintes de uma PLL (Figura 3.5). O sinal de referência e o sinal do VCO são comparados num detector de fase sendo gerado à saída um sinal de controlo em tensão correspondente à diferença das fases. O filtro de malha trata, de seguida, de remover as componentes indesejadas que são produzidas no comparador de fase e, por fim, o sinal de erro de fase irá atacar o VCO. A frequência do VCO, numa PLL bem dimensionada, será igual à

frequência de referência embora os sinais possam ter uma diferença de fase não nula a qual dependerá da natureza dos filtros e do desvio de frequência do VCO em relação à frequência central.

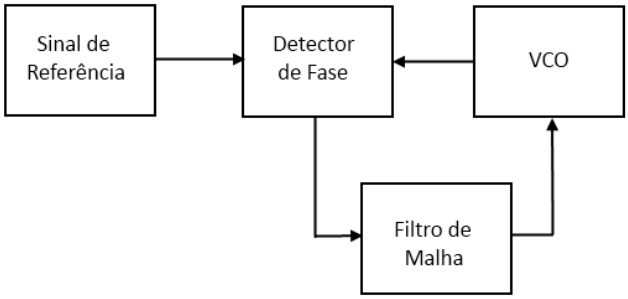


Figura 3.5: Diagrama de blocos de uma PLL básica.

Uma PLL básica é constituída pelos seguintes elementos:

- Detector de Fase;
- Filtro de Malha;
- VCO (Voltage Controlled Oscillator).

O detector de fase é um circuito electrónico desenhado para comparar dois sinais RF, gerando à sua saída uma tensão DC que é função da diferença das fases dos dois sinais. Os detectores podem ser divididos em dois tipos: detector de fase e detector de fase-frequência. Os primeiros são apenas sensíveis a diferenças de fase. Um dos seus problemas reside no facto de que se a diferença de frequências for muito superior à largura de banda do filtro de malha o sinal de controlo nunca chegará ao VCO e o sistema não poderá sincronizar facilmente. Os detectores de fase-frequência têm um funcionamento diferente. Se a diferença de fase estiver entre $\pm 180^{\circ}$, é gerado um sinal à saída em tensão proporcional à diferença de fase dos sinais, tal como o detector de fase. Se a diferença de fase ultrapassar os limites referidos, o detector irá fixar-se num dos seus extremos de forma a ser possível a passagem do sinal de controlo pelo filtro de malha até ao VCO e assim conseguir alcançar o sincronismo.

O filtro de malha tem como objectivo filtrar as componentes de alta frequência que poderão resultar no processo de comparação no detector de fase. Este filtro irá ditar a velocidade com que o sistema muda a frequência. Se o filtro tiver uma frequência de corte baixa, o sistema será lento pois as constantes de tempo serão maiores. Se, pelo contrário, o filtro tiver uma frequência de corte elevada, o sistema irá variar a sua frequência rapidamente. No entanto há algo a notar: com uma frequência de corte elevada irão passar componentes elevadas de frequência que são indesejáveis e que afectarão o desempenho do sistema. É necessário, então, encontrar um compromisso na largura de banda do filtro. O filtro de malha é ainda responsável pelas características do ruído no VCO.

O VCO é um oscilador capaz de ver a sua frequência de oscilação alterada através da aplicação de uma tensão de controlo num *varicap*. A caracterização de um VCO faz-se recorrendo a alguns parâmetros entre eles, a gama de frequências de oscilação, gama de tensão de controlo e o ruído de fase. Apesar ser possível o dimensionamento e

construção de um VCO de raiz para um sintetizador ou PLL, estão disponíveis opções no mercado que apresentam diferentes características consoante o projecto a desenvolver.

O ruído de fase do VCO numa PLL depende da largura de banda da PLL, do ruído de fase do oscilador de referência e do ruído introduzido pelos blocos de *hardware* do sintetizador.

3.2.2 Sintetizador de Frequência baseado em PLL

Um sintetizador de frequência é um circuito capaz de gerar múltiplas frequências partindo de uma ou mais frequências de referência. Os sintetizadores podem ser baseados no *switching* e *mixing* de frequências de um conjunto de osciladores a cristal ou então podem funcionar sobre os princípios de uma PLL, como é o caso do sintetizador que se pretende construir. Estes circuitos são utilizados nas mais variadas aplicações de rádio frequência. Os telemóveis, televisões, rádios ou geradores RF são exemplos onde podemos encontrar estes sintetizadores.

Os sintetizadores baseados em PLL são largamente utilizados no desenvolvimento de osciladores locais para *up* e *down convertion*. O oscilador sintetizado apresenta elevada estabilidade e precisão pois depende apenas de osciladores de referência com boa qualidade os quais são usualmente osciladores a cristal operando em forno aquecido e estabilizado em temperatura (OCXO). A frequência pode facilmente ser controlada a partir de circuitos digitais, tais como, pequenos microprocessadores. Isto possibilita a modificação da frequência de saída em tempo real e com uma elevada resolução.

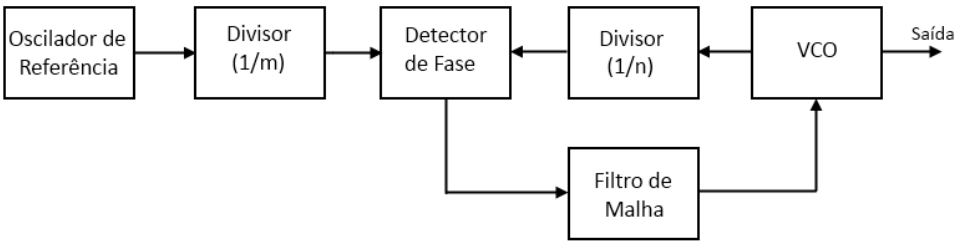


Figura 3.6: Diagrama de blocos de um sintetizador básico.

Um exemplo de um sintetizador básico pode ser encontrado no diagrama de blocos da Figura 3.6. Para transformar uma PLL num sintetizador de frequências, um circuito adicional terá de ser incluído entre o VCO e o detector de fase. Este circuito é o divisor 1/n apresentado no diagrama. O divisor poderá ser fixo ou programável e a frequência sintetizada será múltipla da frequência de referência que habitualmente é 10 MHz.

Para conseguir um sintetizador com uma resolução de frequência mais fina deverá incluir-se um divisor de frequência após o oscilador de referência. Neste caso a frequência à saída da PLL será dada pela expressão (3.6).

Olhando agora para o projecto em concreto, na construção do oscilador local para a IF_2 é necessário um *chip* de síntese, um VCO e um filtro de malha correctamente dimensionado.

Estima-se que o sinal de *beacon* do satélite terá uma variação de ± 3 kHz. O filtro após a *down convertion* apresenta uma largura de banda de ±7.5 kHz, pelo que será necessária uma resolução bastante fina no ajuste para tentar evitar que o sinal caia fora da banda de passagem do filtro e sofra uma atenuação indevida. Neste caso é assumido um valor de 5 kHz de resolução como um valor aceitável. O sinal do oscilador deverá ter uma frequência de 2.0107 GHz com a adição de alguns kHz para compensar as variações de frequência sofridas pelo sinal ao longo do dia. Das arquitecturas que existem para a construção das PLLs a *fractional N* [22] [23] é aquela que se enquadra dentro dos requisitos, portanto os componentes do oscilador local estão dimensionados para a referida arquitectura.

O *chip* do sintetizador é um ADF4153 da Analog Devices [24]. Este chip tem um detector de fase-frequência de baixo ruído, uma *charge-pump* de precisão e um contador *fractional N* programável. A programação poderá ser feita através da escrita nos seus registos utilizando o protocolo SPI.

Quanto ao VCO é um ROS-2015 da Mini-Circuits [17]. A escolha de um VCO depende inicialmente das especificações do *chip* do sintetizador, nomeadamente a frequência de trabalho, a tensão de controlo (que neste caso está entre os 0 e os 5V), o ruído de fase e a sensibilidade. O modelo em questão tem um baixo nível de ruído, baixa sensibilidade e está desenhado para funcionar entre os 1.975 e os 2.015 GHz.

O filtro de malha é bastante importante no design da PLL, uma vez que dita a velocidade de sincronismo e a quantidade de espúrias que poderão passar para o VCO consoante a sua largura de banda. De acordo com as características do sistema e de forma a minimizar os problemas do ruído e das espúrias, o filtro a utilizar será um filtro de 3ª ordem realizado com componentes passivos e dimensionado para uma largura de banda da PLL de 10 kHz. A largura de banda assume habitualmente um valor que é a frequência a que o ruído de fase do VCO se torna inferior ao da referência. Dentro da largura de banda da PLL o VCO é disciplinado pela realimentação para assumir um ruído de fase semelhante ao da referência no detector de fase [25].

$$f_{vco} = n\left(\frac{f_{ref}}{m}\right) \tag{3.6}$$

3.2.3 Oscilador de Referência

O funcionamento de uma PLL ou de um sintetizador não irá depender apenas dos circuitos electrónicos que o constituem, mas também de um sinal de referência externo com uma boa qualidade espectral.

Existem diversos tipos de osciladores de referência que podem ser utilizados neste âmbito, no entanto os osciladores a cristal são dos mais usados pelas suas boas características. Um oscilador a cristal apresenta um factor de qualidade elevado, uma boa estabilidade de frequência e um ruído de fase baixo.

Os osciladores a cristal funcionam com base nas características piezoeléctricas dos cristais. Um campo eléctrico provoca uma deformação mecânica no cristal que se traduz numa vibração de frequência constante.

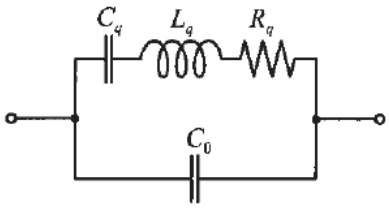


Figura 3.7: Esquema eléctrico de um oscilador a cristal [26].

Dentro da gama dos osciladores a cristal existem tipos diferentes de osciladores que tentam melhorar ainda mais a estabilidade de frequência. Existem, por exemplo, os TCXO (*Temperature Compensated Crystal Oscillators*) que implementam mecanismos de compensação de temperatura ou os OCXO (*Oven Controlled Crystal Oscillators*) que integram um "forno" onde o cristal é mantido a uma temperatura constante.

Os osciladores a cristal têm custos relativamente elevados especialmente para frequências não standard. Por esta razão, no desenvolvimento de um projecto de um receptor ou de um transmissor RF é costume utilizar-se um oscilador a cristal de bastante qualidade e derivar deste outras frequências necessárias, em vez de adquirir vários osciladores de referência. No receptor a construir esta ideia será respeitada e então será utilizado único oscilador de referência para a conversão das duas IF's do sistema.

O oscilador a cristal usado, pelo menos na fase de testes, é um OCXO da MITEQ [27], com referência XTO-05-110.9373-G-15P. Este oscilador tem um baixo ruído de fase e estabilidade de frequência e está dimensionado para produzir um sinal com uma frequência de 110.9373 MHz e uma potência de 11 dBm. Provavelmente no receptor final será adquirido um oscilador standard de 10 MHz que será usado para derivar o primeiro oscilador do receptor. Em termos do projecto implicará programar o sintetizador com outros valores para os divisores.

3.3 Hardware Digital

Os últimos subcapítulos fizeram uma breve descrição do *frontend* analógico que é necessário para que seja possível uma detecção do sinal de *beacon* de satélite num kit rádio digital. Iremos agora apresentar o *hardware* que será utilizado para a implementação do detector digital.

3.3.1 USRP

O USRP é um kit constituído por FPGA, ADC's/DAC's, etc dedicado ao processamento digital de sinal para *software defined radios* desenvolvido pela Ettus Research [28] e agora bastante usado na indústria e no meio académico. Este permite desenvolver sistemas de rádio bidireccionais de baixo custo utilizando um computador pessoal e software compatível com esta plataforma: o GNU Radio. Os principais pontos do GNU Radio irão ser discutidos no próximo capítulo.

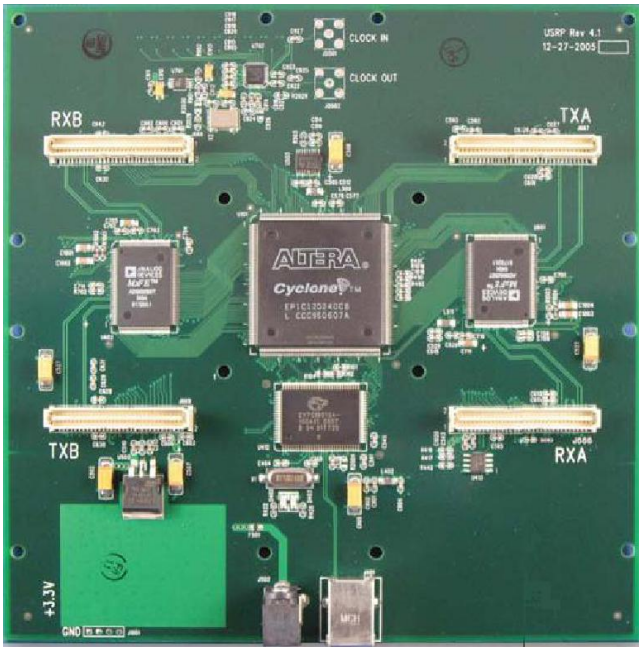


Figura 3.8: USRP *Motherboard***.**

A placa-mãe do USRP é constituída por quatro ADC's e quatro DAC's de alta velocidade, vários portos I/O auxiliares analógicos e digitais e uma FPGA. São ainda incluídos quatro módulos de expansão para placas adicionais: duas para transmissão e outras duas para recepção [29].

Os ADC's de alta velocidade têm uma taxa de amostragem de 64 MS/s e 12 bit de resolução. Para transmissão, os DAC's de alta velocidade funcionam a uma taxa de 128 MS/s e 14 bit de resolução o que leva a uma conversão com uma largura de banda de 64 MHz. Tanto os ADC's como os DAC's de alta velocidade têm disponível um amplificador de ganho programável (PGA) com uma gama de 20 dB.

Para conseguir um melhor aproveitamento da largura de banda e uma maior flexibilidade, as quatro entradas e as quatro saídas dos ADC'c/DAC's poderão ser utilizadas aos pares e assim realizar uma amostragem complexa em vez de uma amostragem real. Desta forma passamos a ter duas entradas complexas (IQ) e duas saídas complexas.

Os portos de entrada e saída analógica estão ligados a ADC's e DAC's de baixa velocidade, respectivamente, e permitem realizar funções como a leitura da temperatura de um circuito externo no caso dos portos de entrada, ou a geração de uma tensão de controlo para um amplificador regulável no caso dos portos de saída. Estão disponíveis 8 portos analógicos para entrada e 8 portos analógicos para saída. Os portos de I/O digitais, 64 bit, são programáveis por *software* e podem ser configurados para funcionarem como portos de entrada ou de saída. Alguns destes portos são utilizados em funções suportadas por *daughterboards* que poderão ser adicionadas posteriormente nos *slots* de expansão.

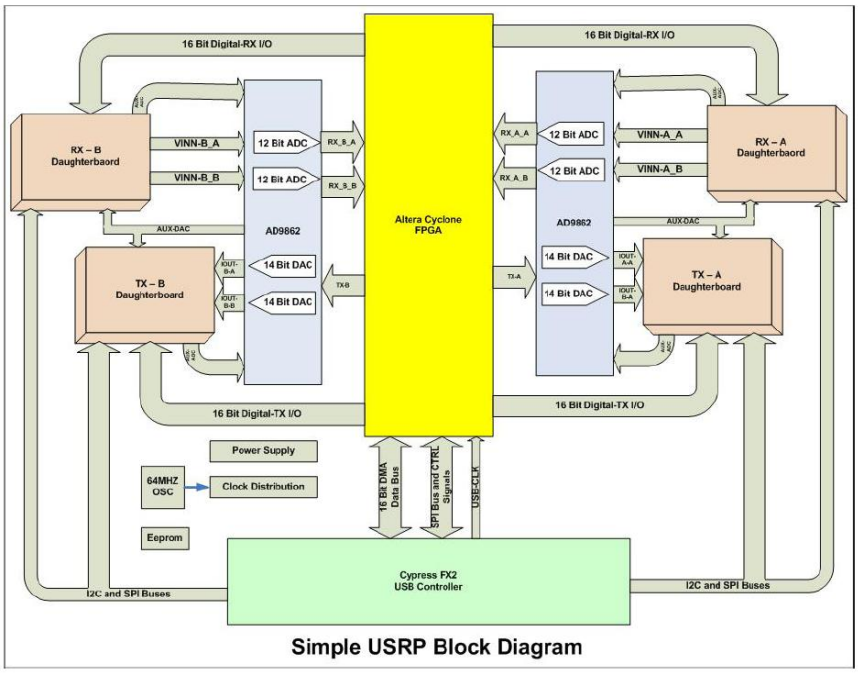


Figura 3.9: Diagrama de blocos do USRP.

A parte mais importante do *hardware* é a FPGA. A FPGA irá efectuar o trabalho mais pesado no processamento do sinal, realizando operações como a *upconvertion* e *downconversion* digital do sinal ou até a decimação e interpolação. A FPGA é ainda responsável pelo envio dos dados para o computador através de uma interface USB 2.0. Tanto os circuitos relativos à FPGA como o controlador da interface USB são programáveis por *software*.

Como exemplo, e sendo algo utilizado na construção no detector, podemos fazer uma breve descrição da *down convertion*. Primeiramente, o sinal sofre uma conversão de frequência da IF para a banda base. O sinal de referência utilizado na conversão é conseguido graças a um NCO (*Numerically Controlled Oscillator*) incluído no USRP. De seguida o sinal é decimado de forma a poder ser transmitido pela interface USB 2.0. O elemento decimador será um filtro passa-baixo seguido de um *down sampler*. O filtro irá deixar passar a banda de interesse, que de seguida será reduzida de um factor N pelo *down sampler*. Após os passos referidos, o sinal irá entrar no computador e a partir daqui o programador poder efectuar qualquer acção utilizando *software*.

3.3.2 USRP *Daughterboards*

A possibilidade de expansão através de placas adicionais torna o USRP uma ferramenta ainda mais interessante para a construção de rádios definidos por *software*. Estas placas permitem a recepção ou a transmissão de um sinal RF e estão disponíveis em várias gamas de frequência para serem utilizadas nas mais diversas aplicações.

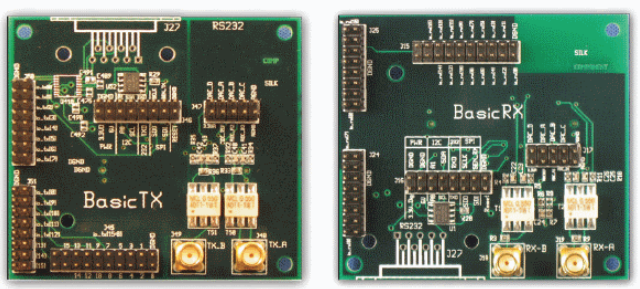


Figura 3.10: USRP *Daughterboards* Basic Tx e Basic Rx.

A Figura 3.10 mostra dois exemplos das *daughterboards* disponíveis para o USRP. A placa Basic Tx é utilizada para transmissão e a Basic Rx tem como fim a recepção. Para o projecto que estamos a desenvolver iremos apenas tirar partido da placa Basic Rx.

A placa Basic Rx está preparada para receber sinais numa gama entre os 0.1 e os 300 MHz. Esta possui duas entradas para sinais RF e está desenhada para levar o sinal RF à placa-mãe do USRP usando apenas transformadores que fornecem um sinal diferencial. A *daughterboard* Basic Rx possui ainda pinos adicionais que permitem a comunicação com alguns dispositivos externos. São suportados os protocolos de comunicação SPI e I^2C que podem ser utilizados para realizar certas funções em conjunto com a recepção dos sinais RF. O sintetizador do receptor tem a possibilidade de ser programado pela interface de comunicação SPI. Ao ser suportado este protocolo de comunicação não será necessária a utilização de outro *hardware* que não o USRP e a *daughterboard* Basic Rx para realizar toda a programação do receptor de propagação.

O kit pode ser utilizado não apenas com uma etapa final de conversão de frequência como ainda assumir outras funções no *hardware* de RF e até eventualmente fazer o controlo de dispositivos externos tal como monitorização de fontes de alimentação, controlo de temperatura do *hardware* ou aquisição de dados meteorológicos (temperatura, pressão, humidade e outros). Esta é a vantagem sobre o *hardware* desenvolvido na UA para estes efeitos.

4 Detector Digital

Agora com um conhecimento global do *hardware* do sistema pela descrição que foi feita nos pontos anteriores, iremos abordar a segunda parte do receptor. Este capítulo será dedicado ao processamento digital de sinal.

Para atingir os nossos objectivos será necessário ter um conhecimento de algumas técnicas de digitalização bem como alguma instrução sobre *software* utilizado. Aqui será feita uma descrição de conceitos generalizados do processamento digital de sinal, dos algoritmos empregados e do *software* utilizado como ferramenta de trabalho.

Como foi já mencionado nos objectivos desta dissertação, a parte relativa à detecção irá continuar o trabalho descrito em [15]. No entanto, e apesar do detector manter a sua estrutura básica, a explicação que aqui será dada não tratará do funcionamento global desta parte do receptor, como foi feito no capítulo 3 referente à explicação do *frontend* analógico, mas sim dos tópicos que servem de base à implementação do detector. Uma vez que existem modificações e adições significativas no *software*, a explicação do seu funcionamento irá ser feita aquando o relato da construção do receptor no capítulo 5.

4.1 Processamento Digital de Sinal

O processamento digital de sinal é utilizado em diversas aplicações que utilizamos regularmente. Operações como o tratamento de uma imagem, compressão de dados ou conversão de um ficheiro de áudio são relativamente vulgares e dependem de técnicas de processamento de sinal adequadas a cada situação.

Processamento digital de sinal não é mais que um conjunto que engloba a matemática, os algoritmos e as técnicas utilizadas para trabalhar um sinal digitalizado. O processamento digital de sinal tem vindo a assumir um papel cada vez mais importante em diversas áreas [30]. Importa salientar:

- Na medicina para diagnósticos através do processamento de dados e imagens resultantes de vários equipamentos médicos;
- Na área comercial com a compressão imagem e vídeo ou produção de efeitos especiais para fins cinematográficos;
- A nível militar com o uso de radares, sonares e comunicações seguras;
- No mundo industrial na monitorização e controlo ou design e teste;
- Na exploração do espaço, com a possibilidade de tratamento de imagem ou compressão de dados.

Naturalmente que esta expansão se deve em grande parte ao aumento da capacidade de processamento que a electrónica veio trazer. No caso particular do receptor a implementar neste trabalho, as técnicas de processamento digital de sinal serão utilizadas para conseguir realizar um detector para um sinal de *beacon* de satélite que oferece algumas vantagens sobre o tradicional detector síncrono o qual, ao exigir a sincronização

do sinal com uma PLL, fica sujeito a limitações devido ao funcionamento precário em condições de baixa CNR. Iremos começar pela explicação dos conceitos iniciais para depois partir para os algoritmos utilizados na obtenção das características do sinal.

4.1.1 Digitalização

Um dos pontos mais importantes na construção de um rádio está na escolha dos elementos que fazem a conversão do analógico para digital, ou vice-versa, ou seja, os ADC's e os DAC's. Estes componentes irão influenciar directamente a gama dinâmica, a largura de banda, a potência consumida e o custo total do sistema, pelo que terá de ser encontrado um compromisso entre os pontos referidos na sua escolha.

O ADC tem como função transformar um sinal analógico, contínuo em termos de amplitude e tempo, num sinal digital discreto em amplitude e tempo. Esta conversão é feita a dois passos com as operações de amostragem e quantização, onde no primeiro é realizada a discretização do sinal no tempo e no seguinte a discretização do sinal em amplitude [12].

Para garantir uma digitalização correcta do sinal é necessária uma frequência de amostragem adequada. Este facto remete para a chamada teoria da amostragem que comummente também é chamada de teorema de Shannon ou teorema de Nyquist. O teorema de Nyquist diz que um sinal não perde informação se for amostrado a uma taxa superior ou igual ao dobro do máximo conteúdo em frequência.

Se existirem componentes de frequência acima da frequência de Nyquist irá ocorrer o chamado *aliasing*, fenómeno que está reproduzido na Figura 4.1. A potência acima da frequência de Nyquist irá sobrepor-se na gama de frequências até Nyquist.

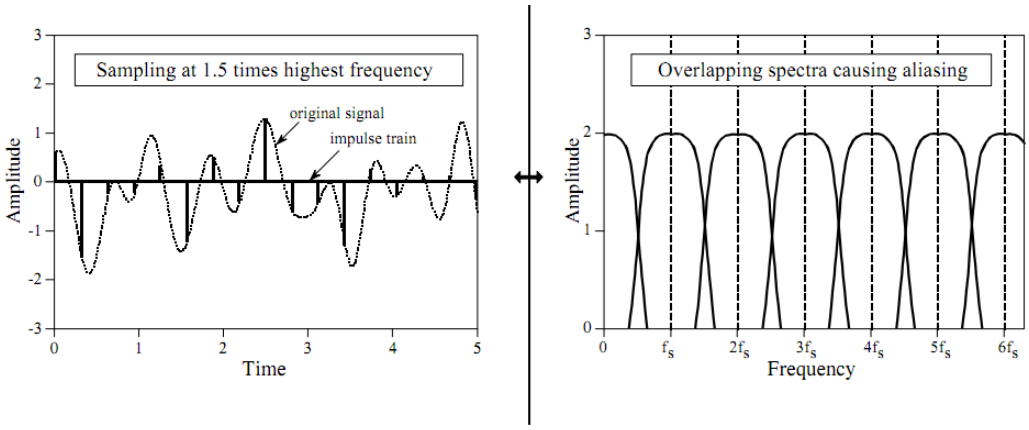


Figura 4.1: Problema de *aliasing* devido ao baixo ritmo de amostragem [30].

Relativamente à discretização do sinal em amplitude, esta irá depender do número de níveis disponíveis, da gama de tensões suportada à entrada e da largura dos níveis de quantização (*step size*). Estes valores são relativos ao ADC escolhido e às suas características. Por exemplo, se o ADC tiver 8 bits, o número de níveis de quantização será dado por 2^B, onde B será 8.

Na selecção de um ADC devem ser levadas em conta as suas especificações. Apesar do objectivo ser sempre a digitalização de um sinal, os ADC's têm diversas arquitecturas.

Numa aplicação de RF, como é o caso do receptor deste trabalho, os ADC's têm uma construção que lhes concede uma elevada velocidade e desempenho. Tipicamente estes ADC's têm 12 bits de resolução.

A resolução e a taxa de amostragem do ADC são geralmente as primeiras especificações que se consultam. No entanto existem outros parâmetros que podem ser igualmente importantes: o *Differential Nonlinearity Error* (DNL), *Signal-to-Noise and Distortion* (SINAD), *Effective Number of Bits* (ENOB) ou até a própria SNR.

Num ADC ideal, um determinado nível de codificação estaria espaçado do seu vizinho o correspondente a 1 LSB (*least significant bit*). O DNL corresponde ao desvio do referido valor. Se este desvio for elevado poderemos perder níveis de codificação.

A SINAD pode ser vista como a relação entre o valor RMS do sinal de entrada e a soma de todas as componentes espectrais abaixo da frequência de Nyquist incluindo as harmónicas, mas excluindo o DC, que irão constituir o ruído. Este parâmetro é semelhante à SNR do ADC com a diferença de a SNR não incluir as harmónicas do sinal.

O ENOB irá depender directamente da SINAD. Este irá revelar qual o número de bits de informação que são realmente úteis em função do ruído. Considerando uma sinusóide como sinal de entrada, a ENOB pode ser dada pela expressão (4.1). Este valor tende a deteriorar-se com o aumento da frequência. De acordo com a sinusóide que corresponderá ao sinal que se pretende digitalizar será possível avaliar a performance do ADC em termos de número efectivo de bits.

$$N = \frac{(SINAD - 1.76\,dB)}{6.02} \tag{4.1}$$

Uma outra questão importante no que diz respeito aos ADC's é a resolução relativamente à gama dinâmica esperada para o sinal.

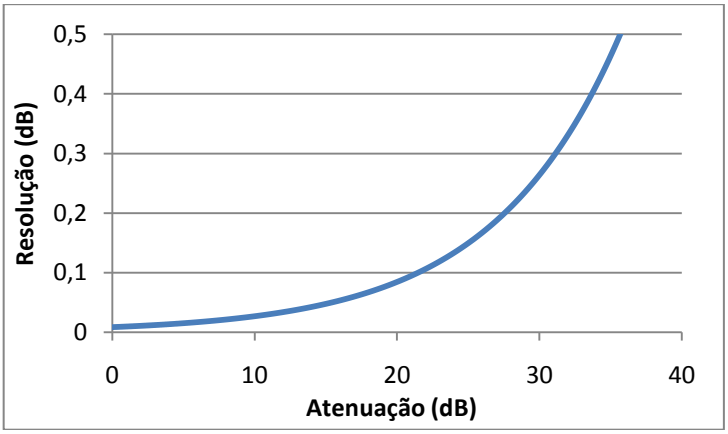


Figura 4.2: Traçado Resolução vs Gama Dinâmica.

Na Figura 4.2 está um gráfico da resolução de um ADC de 12 bit (4096 níveis) admitindo que o sinal de maior amplitude é digitalizado com o nível máximo em função da correspondente atenuação. O traçado do gráfico mostra que à medida que aumentamos a gama dinâmica iremos estar a sacrificar a resolução, o que pode introduzir erros na medição. Vimos anteriormente que era pretendida uma gama dinâmica da ordem de 25 dB para o acompanhamento das variações do sinal do *beacon*. Este valor

corresponde a uma resolução de cerca de 0.15 dB, algo ainda aceitável para cumprir com os requisitos. O sinal *crosspolar* deverá ser amplificado antes da aquisição para beneficiar de uma melhor resolução.

O ADC que irá digitalizar o sinal do *beacon* tem, segundo as características do fabricante, um conversor de 12 bit. Contudo a amostragem de um sinal com ruído de amplitude e largura de banda suficientes a uma frequência mais elevada que o necessário, seguida de uma média, pode aumentar a resolução efectiva e melhorar a resolução [31] [32]. Isto consegue-se com a introdução de *dither* [33].

O *dither* não é mais que ruído pseudo-aleatório que é introduzido intencionalmente na entrada analógica de sistema. Com este ruído adicional irá ser possível como que adicionar novos níveis de quantização. Como exemplo, suponhamos uma ADC apenas com 2 níveis: 0 e 1. Se for introduzido ruído aleatório com uma distribuição uniforme iremos ter o nível 0, o nível 1 e o valor 1 com uma determinada probabilidade. Quando realizada a média do sinal discretizado iremos ter um resultado bastante próximo da média do sinal analógico, mantendo a linearidade.

Este será o princípio utilizado para melhorar a performance do ADC. A questão prende-se agora com o ruído que deverá ser introduzido. Se analisarmos o sinal que será digitalizado podemos verificar que este já é acompanhado por ruído aleatório numa largura de banda que é superior a 10 kHz. Esta largura de banda irá introduzir ruído suficiente (excede a tensão entre dois níveis de quantificação) para que ocorra o aumento da resolução.

4.1.2 Transformada de Fourier

A transformada de Fourier é baseada em algoritmos matemáticos que envolvem a decomposição de sinais em sinusóides e permitem obter a representação de um sinal no tempo no domínio da frequência. Uma vez que os computadores digitais conseguem lidar apenas com sinais discretos e finitos as únicas técnicas adequadas são relativas à transformada de Fourier discreta (DFT).

A DFT real converte um sinal constituído por N amostras no domínio do tempo em $N/2 + 1$ valores complexos no domínio da frequência.

A informação presente no sinal no domínio das frequências é exactamente a mesma contida no sinal descrito no tempo, apenas a sua representação é diferente. Aliás, sabendo o domínio no qual o sinal está representado, é possível conseguir a representação do sinal no outro. A passagem do domínio do tempo para as frequências é simplesmente chamado de transformada de Fourier discreta, o contrário será a transformada de Fourier discreta inversa.

Uma das propriedades mais importantes da transformada de Fourier é a linearidade. Esta propriedade é bastante útil a nível prático, se não olhemos para a Figura 4.3. Nesta figura temos um sinal representado no domínio do tempo que é constituído por diferentes frequências com diferentes amplitudes. Tentar descobrir quais as frequências moduladas no sinal simplesmente olhando para o traçado é uma tarefa bastante complicada no domínio do tempo. No entanto, se o observarmos na representação

equivalente nas frequências facilmente distinguimos as frequências do sinal bem com as amplitudes correspondentes.

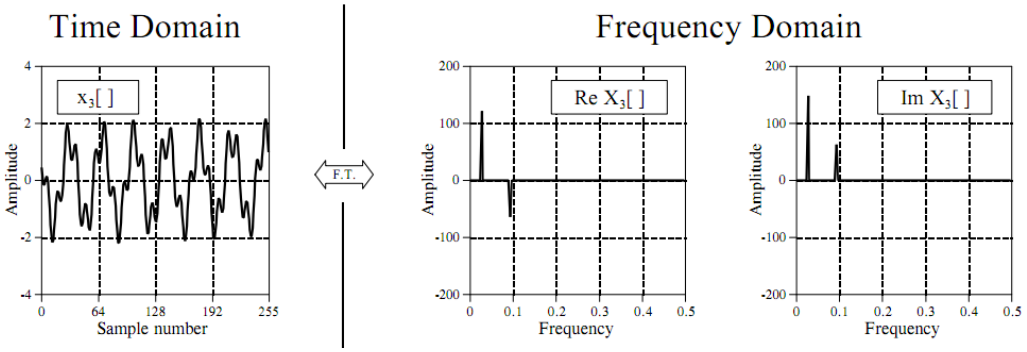


Figura 4.3: Transformada de Fourier – Múltiplas frequências moduladas num sinal [30].

4.1.2.1 Transformada de Fourier Discreta Complexa

A DFT complexa pode ser vista como uma versão mais sofisticada da DFT real. A sua criação tem em vista eliminar alguns problemas e limitações que a DFT real possui.

A Tabela 4.1 mostra uma representação matemática para a DFT real e DFT complexa, bem como para as transformadas inversas [30].

Tabela 4.1: Representação matemática da DFT real e da DFT complexa.

Discrete Fourier Transform (DFT)

complex transform	real transform
synthesis $x[n] = \sum_{k=0}^{N-1} X[k]\ e^{j2\pi kn/N}$	*synthesis* $x[n] = \sum_{k=0}^{N/2} ReX[k]\cos(2\pi kn/N) - ImX[k]\sin(2\pi kn/N)$
analysis $X[k] = \frac{1}{N}\sum_{n=0}^{N-1} x[n]\ e^{-j2\pi kn/N}$	*analysis* $ReX[k] = \frac{2}{N}\sum_{n=0}^{N-1} x[n]\cos(2\pi kn/N)$ $ImX[k] = \frac{-2}{N}\sum_{n=0}^{N-1} x[n]\sin(2\pi kn/N)$
Time domain: x[n] is complex, discrete and periodic; n runs over one period, from 0 to N-1	Time domain: x[n] is real, discrete and periodic; n runs over one period, from 0 to N-1
Frequency domain: X[k] is complex, discrete and periodic; k runs over one period, from 0 to N-1; k = 0 to N/2 are positive frequencies; k = N/2 to N-1 are negative frequencies	Frequency domain: Re X[k] is real, discrete and periodic; Im X[k] is real, discrete and periodic; k runs over one-half period, from 0 to N/2
	Note: Before using the synthesis equation, the values for Re X[0] and Re X[N/2] must be divided by two.

O ponto mais importante a reter da análise da DFT complexa reside na capacidade de distinção de frequências positivas e negativas. Em termos práticos, esta característica irá permitir não só observar o dobro da largura de banda mas também perceber, no caso de uma conversão de frequência, se a conversão está a ser realizada com o oscilador local

acima ou abaixo da frequência de RF. Ao proceder à análise das componentes cartesianas de um sinal convertido em frequência irá ser possível sintonizar adequadamente os osciladores do sistema.

4.1.2.2 Fast Fourier Transform (FFT)

A FFT é um algoritmo eficiente para calcular a transformada de Fourier discreta.

Existem bastantes algoritmos disponíveis que exploram o facto de usarem quase sempre um número de amostras *N* do sinal do tipo que é um múltiplo de 2. Uma DFT pode exigir um número proporcional a N^2 operações aritméticas enquanto uma FFT exige um número proporcional a $N\log(N)$ o que representa um notável ganho computacional para um elevado número de amostras *N*. Dos vários métodos que realizam a FFT, o método radix-2 Cooley Tukey é um dos mais populares e mais utilizados [34].

Seja qual for o método para calcular a FFT, o seu princípio baseia-se na decomposição de um sinal de *N* amostras em sucessivos segmentos com um número de amostras menor.

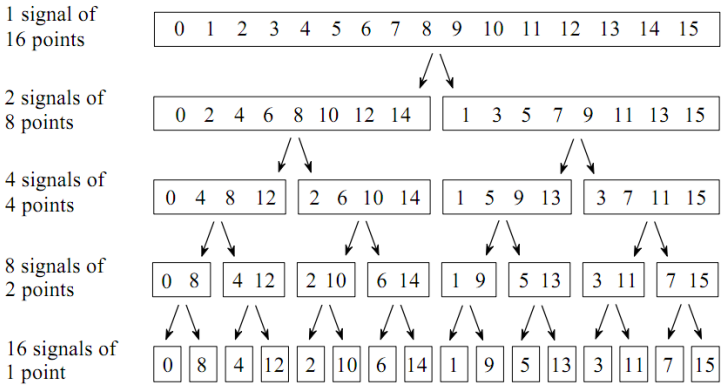


Figura 4.4: FFT- Exemplo de decomposição de um sinal [30].

A Figura 4.4 dá um exemplo do primeiro passo na realização da FFT que envolve uma decomposição. Esta decomposição irá resultar numa reorganização das amostras do sinal onde, num passo seguinte, cada amostra será descrita pelo seu espectro de frequência. A síntese das amostras num único espectro será a última etapa deste processo e esta será a parte que irá exigir um maior esforço computacional por ser necessária a inversão passo-a-passo das divisões feitas inicialmente.

A criação de um algoritmo para realizar uma FFT poderá não ser algo trivial mas felizmente existem vários programas que os incluem nas suas bibliotecas para uso imediato. Tanto o *software* Matlab [35] como o GNU Radio incluem estes algoritmos que podem ser utilizados através da simples invocação de um comando ou procedimento.

4.2 Algoritmos para Caracterização do Sinal

Abordamos aqui os métodos usados para, a partir do espectro das componentes cartesianas do sinal adquirido, conseguir efectuar uma estimativa da frequência (eventualmente útil para programar algum oscilador da cadeia) e da potência ou amplitude do sinal *copolar* e *crosspolar* e ainda da respectiva fase relativa.

4.2.1 Frequência

Usando o espectro podemos localizar o índice do *bin* de maior amplitude e determinar a frequência do sinal contudo tal poderá não ser suficiente. Se não vejamos, um sinal não tem toda a sua potência numa frequência mas sim numa largura de banda. Devido ao ruído de fase que acompanha o sinal, a sua potência está espalhada numa largura de banda de algumas dezenas de Hz.

Quando é realizada a FFT a potência do sinal irá ser dividida em *bins*. Portanto a frequência média poderá não corresponder à do *bin* de maior amplitude. Para calcular com maior rigor a frequência do sinal a partir do espectro existem várias alternativas. Uma delas passa pela utilização de uma resolução mais fina de frequência que leva a um esforço computacional maior e uma colecção de dados mais longa temporalmente. Como foi referido anteriormente a FFT ainda exige um esforço computacional considerável. Uma solução passa pela realização de uma média pesada das frequências dos *bins* de maior potência [36]. Esta solução será a utilizada no detector para uma melhor estimativa de frequência.

$$f_c = \frac{\sum_{i=1}^{n_{bins}} (bin_i \,.\, bin_i^* \,.\, f_{bin_i})}{\sum_{i=1}^{n_{bins}} (bin_i \,.\, bin_i^*)} \tag{4.2}$$

A fórmula anterior, onde o * significa complexo conjugado, realiza a estimativa pesando a frequência de cada *bin* pela respectiva potência.

O maior problema nesta solução está na largura de banda a considerar em torno do *bin* de maior amplitude. O que se pode verificar é que se a relação CNR for muito baixa será preferível usar um número reduzido de *bins*. Os *bins* mais afastados da frequência central terão uma potência de sinal reduzida e consequentemente severamente alterada pelo ruído branco (uma maior variância na estimativa de frequência). No *software* a largura de banda a considerar é dada como parâmetro de entrada.

4.2.2 Potência do Sinal *Copolar*

Tal como a determinação da frequência, a estimativa da potência dos sinais *copolar* e *crosspolar* é um dos objectivos principais do trabalho.

Uma largura de banda mínima contida n_{bins} em volta do *bin* de maior amplitude deverá ser considerada para a determinação da potência do sinal. A potência pode então ser estimada com a soma das potências dos *bins* de interesse:

$$P_{sinal} = \sum_{i=1}^{n_{bins}} (co_i \, . \, co_i^*) \tag{4.3}$$

E a amplitude será:

$$A_{co} = \sqrt{P_{sinal}} \tag{4.4}$$

Uma vez que a potência do sinal recebido é variável, o número de *bins* utilizado para o cálculo em (4.3) bem como a sua resolução irão influenciar os resultados. A utilização de uma resolução mais fina apresenta sempre melhores resultados contudo devemos ter em conta que a medição dos sinais exige uma resolução temporal mínima e consequentemente o período temporal em análise é sempre limitado (detalhar finamente o espectro exige um longo tempo). Nesta situação, o número de *bins* utilizado para a obtenção do valor da potência do sinal irá depender da CNR. Se o patamar de ruído estiver muito abaixo do nível do sinal, a utilização de um maior número de *bins* não irá alterar significativamente a estimativa da amplitude do *copolar*. Mas se pelo contrário a CNR for reduzida, um número de *bins* maior provoca um desvio médio sistemático positivo pois a potência de ruído começa a ser significativa relativamente à do sinal.

O desvio sistemático no *copolar* é dado por:

$$\Delta P_{sinal} = 10 \log\left(1 + \frac{1}{SNR}\right) \tag{4.5}$$

com a SNR (calculada na largura de banda de detecção) dada em valores lineares.

Podemos pensar no domínio do tempo. Se uma FFT for realizada sobre um número elevado de amostras (resolução elevada de frequência) há uma integração implícita no tempo: a variância da estimativa do sinal é menor e a resolução temporal mais grosseira o que pode prejudicar a medição da dinâmica dos fenómenos de propagação. Se usarmos um período mais curto para adquirir as amostras as estimativas de amplitude apresentarão uma maior variância contudo teremos uma maior resolução no tempo. A média das estimativas neste último caso, quando estendida a um tempo equivalente ao primeiro, apresentará uma variância semelhante.

Para resolver esta questão um mecanismo para o controlo da largura de banda na estimativa da potência do sinal e a possibilidade de configuração na fase inicial serão introduzidos no *software* do detector.

4.2.3 Amplitude da Componente Despolarizada *Crosspolar* e Fase Relativa

A determinação da amplitude do sinal *crosspolar* não pode ser realizada de forma directa como no caso do *copolar* pois a SNR do *crosspolar* é de forma geral bem menor que a do *copolar* e, nestes casos, a estimativa rapidamente saturaria devido à potência do ruído. Podemos contudo explorar o facto de os sinais serem coerentes.

A solução pensada tem um princípio relativamente simples. Em termos práticos são seleccionados o *bin* do *copolar* de maior amplitude e alguns adjacentes que contêm a potência do *copolar* bem como os *bins* correspondentes do sinal *crosspolar*. É feita uma multiplicação complexa, índice a índice, dos *bins* do *copolar* e do *crosspolar* e o respectivo

somatório. O valor obtido $(a + jb)$ será correspondente à multiplicação do módulo do vector *CO*, com o módulo do vector *CX* e com a fase relativa θ.

$$\sum_{i=1}^{n_{bins}} (cx_i . co_i^*) = (a + jb) \rightarrow |CO|.|CX|.(\cos\theta + j\sin\theta) \tag{4.6}$$

A divisão, em valor absoluto, deste somatório pela amplitude do sinal *copolar* resulta no valor da amplitude do *crosspolar* pretendida.

$$A_{cx} = \frac{\sqrt{a^2 + b^2}}{A_{co}} \tag{4.7}$$

Relativamente à fase relativa entre os sinais, esta pode ser agora obtida directamente com a utilização da função arctan2. Como parâmetros da referida função, são utilizados os valores real e imaginário do somatório dos produtos dos *bins* do sinal *copolar* com os *bins* correspondentes do sinal *crosspolar*.

$$\theta = \arctan 2\,(b, a) \tag{4.8}$$

Esta solução acaba por reduzir os erros de medida sistemáticos de amplitude e aleatórios da fase de uma forma considerável pois a efectiva largura de banda da detecção do *crosspolar* é a resolução espectral da FFT.

4.2.4 NSD e CNR

A NSD (densidade espectral de ruído) corresponde à potência de ruído por Hertz. A estimativa desta quantidade permitirá averiguar a qualidade do sinal ou seja a CNR. Para determinar o seu valor utiliza-se um método semelhante ao da estimativa do valor da potência do sinal.

Conhecendo a frequência central e a largura de banda ocupada pelo sinal, selecciona-se uma determinada banda acima e abaixo do *bin* com maior amplitude e procede-se à soma dos *bins* correspondentes ao ruído. Dividindo o resultado anterior pela largura de banda ocupada pelos *bins* consegue-se uma estimativa para a NSD. De notar que, como o ruído é aleatório, ao incluir uma largura de banda maior podem obter-se resultados mais precisos desde que a densidade espectral de ruído seja plana no espectro seleccionado.

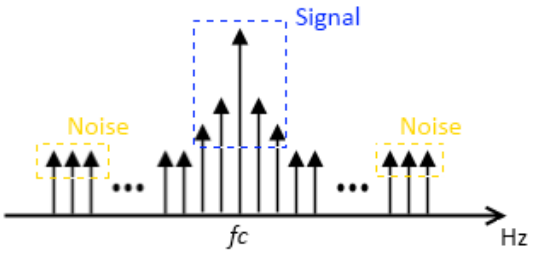


Figura 4.5: Representação do sinal do *beacon* digitalizado.

$$NSD = \frac{\sum_{i=1}^{n_{bins\ abaixo}} (co_i\ .\ co_i^*) + \sum_{i=1}^{n_{bins\ acima}} (co_i\ .\ co_i^*)}{n_{bins\ total}\ .resolution} \tag{4.9}$$

A determinação da CNR é agora imediata. Com a NSD e com a potência do sinal já determinada no ponto 4.2.2 é agora possível determinar o valor da CNR recorrendo à expressão (4.10).

$$CNR = 10\log\left(\frac{P_{sinal}}{NSD}\right) \tag{4.10}$$

4.2.5 Variância do *Copolar*

Além dos parâmetros já mencionados irá ainda ser determinada a variância do *copolar*. Este parâmetro irá permitir uma análise da estabilidade do sinal ao longo do tempo e poderá ser um indicador de turbulência do meio de propagação. Pretende-se apresentar a variância na interface gráfica para uma informação em tempo real e actualizada de 30 em 30 segundos ou de 1 em 1 minuto.

Para o cálculo da variância (dB^2) iremos recorrer a um algoritmo recursivo utilizando as expressões (4.11) e (4.12).

$$\overline{x}_n = \left(\frac{n-1}{n}\right).\overline{x}_{n-1} + \left(\frac{1}{n}\right).x_n \tag{4.11}$$

$$\sigma_n^2 = \sigma_{n-1}^2\,.\left(\frac{n-1}{n}\right) + (x_n - \overline{x}_n)^2\,.\left(\frac{1}{n-1}\right) \tag{4.12}$$

A variância (σ_n^2) irá depender do seu valor passado (σ_{n-1}^2) e da média dos valores das amostras ($\overline{x}_n$). Esta irá ser actualizada a cada nova amostra x_n.

O método apresentado irá apenas manter um histórico do valor actual e do último valor passado da média e da variância. Desta forma são reduzidos a memória e os recursos computacionais utilizados na realização do cálculo.

4.3 Software

Depois de terem sido introduzidos os conceitos de processamento digital de sinal e também os algoritmos de cálculo para determinar amplitudes do *copolar* e *crosspolar* e respectiva fase relativa, resta agora efectuar a implementação.

Os vários pontos desta secção irão descrever o *software* que será utilizado para a concretização do detector digital.

4.3.1 GNU Radio

O GNU Radio pode ser visto como um conjunto de ferramentas de *software* que é dedicado ao processamento digital de sinal para a criação de rádios definidos por *software* de baixo custo. O projecto do GNU Radio foi iniciado por Eric Blossom [37] e as suas bibliotecas podem ser utilizadas de forma gratuita por qualquer pessoa. Existe também uma vasta e dedicada comunidade ligada ao GNU Radio que fornece ajuda em diversos fóruns na internet e assim facilita a resolução de problemas de implementação de forma mais rápida.

Para entender o funcionamento básico do GNU Radio vamos começar com um exemplo simples. Consideremos um simples bloco. Este bloco irá realizar uma qualquer operação de processamento digital de sinal. Agora, para tirar partido da funcionalidade deste bloco vamos incluir um elemento que coloque um sinal digital à entrada do sistema. Este elemento pode também ser representado por um novo bloco que será classificado como *source*. Precisamos agora de algo para terminar o sistema de forma conveniente. Um último bloco, do tipo *sink*, é adicionado e temos por fim um sistema funcional.

Figura 4.6: Diagrama de blocos de um sistema GNU Radio básico.

Este é o paradigma do GNU Radio. Um sistema é implementado à custa de uma rede de blocos interligados de forma conveniente que realizam as operações desejadas. Esta arquitectura é utilizada em diversas aplicações de *software* como, por exemplo, a ferramenta Simulink, incluída no Matlab, que permite analisar a controlabilidade de um sistema através da sua construção teórica num diagrama de blocos.

No GNU Radio temos, então, blocos e ligações. Estes elementos resultam da fusão de duas linguagens de programação: o Python e o C++.

Todos os blocos incluídos no GNU Radio são implementados em C++, ou seja, todo o trabalho crítico de processamento do sinal é realizado em C++. Isto faz todo o sentido uma vez que o C++ apresenta um bom desempenho e além disso é uma linguagem já com alguma história, sendo bem conhecida pelos programadores. Os blocos possuem como atributos o número de portas de entrada, o número de portas de saída e o tipo de dados que tratam (short, float, etc). Estes podem ser um de três tipos: *sources*, *sinks* ou blocos de processamento de sinal. Os blocos do tipo *source* situam-se na entrada do sistema e introduzem o sinal. Os de tipo *sink* terminam o diagrama e representam o destino final do sinal (ficheiro, interface gráfica, etc). Os blocos de processamento digital realizam as operações para as quais foram desenhados. Como exemplos destes blocos, podemos mencionar um bloco *source* que faz a leitura de uma ADC, um bloco *sink* que escreve os dados de um sinal para um ficheiro e um bloco de processamento de sinal que realiza um qualquer filtro.

O diagrama do sistema, também chamado de *flow graph*, é construído com a linguagem de programação Python. O Python é uma linguagem de alto nível, com uma sintaxe simples e orientação a objectos [38]. Esta linguagem permite o desenvolvimento

de uma aplicação de forma rápida, sem a necessidade de um longo período de aprendizagem. O código Python pode ser lido facilmente e o seu *debug* não apresenta tantas complicações como, por exemplo, o C++. De facto, um programa escrito em Python não necessita de ser compilado antes da sua execução. O *debugger*, quando o programa é executado, verifica se existe algum erro ou falha. Se existir é levantada uma excepção que serve como referência para o programador corrigir o problema. Com isto, o desenvolvimento e teste torna-se mais simples e mais rápido. Uma outra particularidade interessante do Python é o facto de incluir a capacidade de ligar o código C++. Isto é o que na prática acontece no GNU Radio. O *flow graph*, como já vimos, é constituído por blocos e ligações. O Python irá permitir a conexão dos vários blocos construídos em C++. Além disto o Python será ainda responsável pela configuração e controlo de todo o sistema.

Para uma melhor percepção do funcionamento do GNU Radio, vamos proceder à explicação do chamado "*Hello World*" desta plataforma de *software*.

```
1: #!/usr/bin/env python
2: from gnuradio import gr
3: from gnuradio import audio
4:
5: def build_graph ():
6:   sampling_freq = 48000
7:   ampl = 0.1
8:
9:   fg = gr.flow_graph ()
10:          src0 = gr.sig_source_f (sampling_freq, gr.GR_SIN_WAVE,
   350, ampl)
11:          src1 = gr.sig_source_f (sampling_freq, gr.GR_SIN_WAVE,
   440, ampl)
12:          dst = audio.sink (sampling_freq)
13:          fg.connect ((src0, 0), (dst, 0))
14:          fg.connect ((src1, 0), (dst, 1))
15:          return fg
16:
17:  if __name__ == '__main__':
18:          fg = build_graph ()
19:          fg.start ()
20:          raw_input ('Press Enter to quit: ')
21:          fg.stop ()
```

Este exemplo é chamado de "*Dial Tone Output*" e é um dos vários exemplos que acompanha o pacote do GNU Radio. A sua função é reproduzir um som semelhante ao tom que é característica de um telefone quando se pretende realizar uma ligação. A primeira linha é incluída para que este script possa ser tornado executável de forma directa (não é obrigatório mas é algo frequentemente utilizado). As linhas 2 e 3 dizem respeito à importação das bibliotecas utilizadas pelo programa. Pode ser importado um pacote, um módulo de um pacote ou até apenas uma definição dentro de um módulo. O primeiro módulo, a biblioteca *gr*, contém a maior parte dos blocos presentes no GNU Radio, incluindo o importante *top_block* que serve como base à criação dos *flow graphs*. O módulo *audio* é importado porque a saída do sistema será a placa de som do computador, de forma a ser audível o sinal em análise. Nas linhas 10 e 11 são criados dois blocos que são as *sources* do sistema. Aqui serão geradas duas sinusóides de frequências

350 Hz e 440 Hz. A linha 12 diz respeito ao bloco que termina o diagrama, um *sink* que dirigirá o sinal da sua entrada para a placa de som. Com a criação de uma estrutura para o *flow graph* na linha 9, resta-nos agora ligar os vários blocos do sistema. Este processo é realizado nas linhas 13 e 14 com a ajuda do comando "connect" incluído na biblioteca *gr*. Por fim, para que o *flow graph* entre em funcionamento é necessário ser incluída a instrução "start", como acontece na linha 19. Se o script for executado é possível confirmar que tudo funciona correctamente ao ouvir uma representação sonora do sinal nos altifalantes do computador.

Enquanto o princípio de funcionamento do GNU Radio pode ser considerado simples, a implementação de um sistema poderá não ser tão acessível, principalmente se for pretendida a integração de uma função que não exista nas bibliotecas de blocos do GNU Radio. A necessidade da construção de um novo bloco resulta num acréscimo de dificuldade considerável, uma vez que será necessário interagir com o código C++ para a criação do mesmo. Um novo bloco será desenvolvido à custa de três ficheiros diferentes que, quando compilados, resultam num novo módulo que pode ser importado num script em Python. Estes ficheiros são: um *header* (.h) que contém os cabeçalhos das funções realizadas pelo bloco, bem como as entradas e saídas; um script C++ (.cc) que descreve o trabalho a realizar pelo bloco; um ficheiro SWIG [39] (.i) que permite "colar" o código C++ ao código Python. Uma descrição de como criar um novo bloco pode ser consultada no sítio da internet oficial do GNU Radio [37].

A utilização desta plataforma de *software* torna-se ainda mais interessante com a utilização do USRP. Aliás, o GNU Radio e o USRP apresentam uma cumplicidade que permite juntar os dois recursos para realizar rádios de forma directa. O GNU Radio contém um conjunto de bibliotecas dedicadas ao USRP que permitem tirar partido das suas funcionalidades, sem que seja necessária a implementação de controladores adicionais. A maior dificuldade aqui está mesmo na escassa documentação que existe para consulta.

O GNU Radio está disponível para diversos sistemas operativos, nomeadamente o Linux, Windows e Mac OS, embora seja mais comummente utilizado o Linux, como é o caso do detector a implementar neste trabalho. Guias para a instalação deste *software* podem ser encontrados para cada um dos referidos sistemas operativos na página oficial do GNU Radio [37].

4.3.2 Linguagens de Programação e Bibliotecas Adicionais

Vimos anteriormente que o GNU Radio utiliza Python e C++ para construir rádios. No entanto, recorrendo apenas a estas duas linguagens de programação, não é possível conseguir uma boa interacção com o utilizador. Os próximos pontos irão explicar as ferramentas de *software* adicionais que foram utilizadas para desenvolver a interface gráfica do detector digital.

4.3.2.1 wxPython

Antes de ser explicada a estrutura do wxPython convém fazer referência à forma como este funciona. O wxPython pode ser considerado como uma interface para um *toolkit* criado em C++ chamado wxWidgets, uma ferramenta de *software* para a criação de interfaces gráficas bem conhecida [40]. Tal como o GNU Radio, um código "cola" chamado SWIG é utilizado para que seja possível uma conexão entre o Python e o C++. Assim é possível usufruir das funcionalidades do *toolkit* wxWidgets com a simplicidade de implementação do código Python.

O wxPython permite a criação de interfaces gráficas com uma elaboração mais ou menos complexa dependendo das necessidades da aplicação. As suas funcionalidades podem ser utilizadas para os mais diversos fins, tanto que estão disponíveis versões para Windows, Linux e Mac OS. Os exemplos incluídos no GNU Radio, que envolvem uma *graphical user interface* (GUI), têm como base o wxPython e, como eles, o detector digital fará uso das suas bibliotecas.

Tal como o Python, o wxPython é orientado a objectos. A criação de uma nova janela irá ter como base uma determinada classe e irá derivar desta. Para entender os conceitos do wxPython comecemos por analisar um programa simples escrito nesta linguagem de programação. Este programa irá criar a janela da Figura 4.7.

```
1: #!/usr/bin/python
2: import wx
3:
4: class GUI(wx.Frame):
5:  def __init__(self, title):
6:          wx.Frame.__init__(self, None, title=title, size=(300,150))
7:          self.Show()
8:          self.Center()
9:
10: if __name__ == '__main__':
11:         app = wx.App()
12:         GUI('Empty Window')
13:         app.MainLoop()
```

Como iremos utilizar as bibliotecas do wxPython necessitamos de as importar para o nosso programa logo à partida. Isto é feito na linha 2 do código. Tal como foi já referido, as novas janelas irão derivar de classes já existentes. Assim, na linha 4, é declarada uma nova classe "GUI" que será do tipo wx.Frame, ou seja, uma janela principal ou *top-level*. Na linha 5 é feita a inicialização da nova classe com a indicação de que esta vai ter um parâmetro de entrada, neste caso, a variável "title". De seguida é necessário inicializar a classe wx.Frame, o que é feito na linha 6. Aqui, os parâmetros de entrada indicam que esta janela não tem uma janela pai, tem o título igual ao conteúdo de "title" e tem uma dimensão dado pelo "size". As linhas 7 e 8 servem apenas para tornar a janela visível e para a centrar no ecrã do computador. Por fim, para que o programa seja executado é necessário recorrer à wx.App. Qualquer programa em wxPython será uma instância do wx.App, portanto, após inicializar a janela na linha 12, utiliza-se o comando "Mainloop()" na linha seguinte para que o programa entre em funcionamento de forma ininterrupta até que um evento o termine.

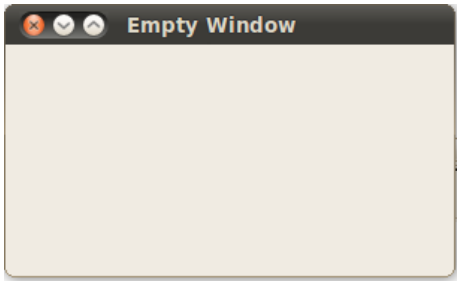


Figura 4.7: Exemplo de uma janela criada em wxPython.

Há pouco foi mencionado o conceito de janela *top-level*. Este é um bom ponto de partida para o entendimento da estrutura de uma GUI criada em wxPython. Uma janela *top-level* alberga todos os outros constituintes da interface. Esta é definida pela classe wx.Frame e nunca se pode colocar uma janela definida por esta classe dentro de uma outra wx.Frame. Para incorrer em boas práticas, normalmente são utilizados *containers* para hospedar os vários objectos que se pretendam incluir. Estes *containers* são criados com a classe wx.Panel, uma derivada da wx.Frame. Como exemplo para uma estrutura de uma janela, olhemos para a Figura 4.8.

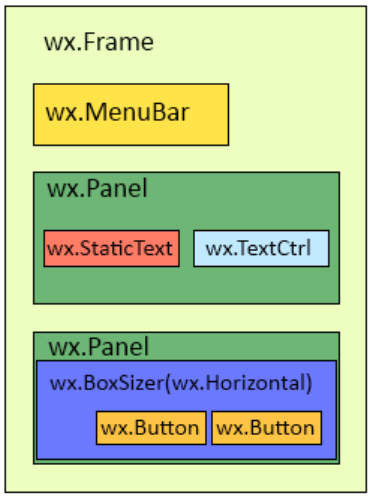


Figura 4.8: Exemplo de um *layout* de uma janela no wxPython.

Este é um exemplo que mostra a estrutura hierárquica praticada no wxPython. A wx.Frame é a classe de mais alto nível, seguido da wx.Panel e posteriormente as subclasses que implementam as funções, como por exemplo uma caixa de texto para realizar uma autenticação. Existem ainda outras subclasses que normalmente são afixadas à janela principal directamente. Este é o caso dos menus criados pela wx.Menubar perto do topo da janela, como é mostrado na Figura 4.8, ou uma barra de estado realizada com a subclasse wx.StatusBar.

Uma outra questão está na disposição dos elementos que constituem a janela. Grande parte das classes que realizam estes elementos possui como parâmetros de entrada a posição e o tamanho. No entanto, quando a janela começa a ter um número alargado de elementos a disposição dos mesmos começa a não ser tão trivial. Uma das formas de organizar a nossa janela passa pela utilização de *boxsizers*. Os *boxsizers* são criados com as classes wx.BoxSizer e podem ter o alinhamento do seu conteúdo na horizontal ou na

vertical. Estes permitem posicionar e dimensionar todo o seu conteúdo da forma que mais nos agradar ou mesmo de forma automática. Na Figura 4.8 encontra-se um exemplo de um *boxsizer* que alinha horizontalmente dois botões no seu interior.

Relativamente à interacção com o utilizador, as várias operações que são permitidas numa janela são realizadas à custa de eventos. Um evento pode ser considerado uma interrupção, ou seja, quando um determinado evento ocorre, o processador que estava a cumprir uma determinada instrução pára ou termina essa instrução e executa a operação associada ao evento. Os eventos que existem numa janela são variados e podem estar associados a um sem número de operações. Estes podem passar por um simples movimento do rato dentro da janela, até a um clique numa das opções de um menu.

A explicação aqui dada trata de uma forma breve os conceitos básicos do wxPython. As suas possibilidades irão depender não só do conhecimento da linguagem mas também da criatividade do programador. Como referências, existem vários livros e vários tutoriais disponíveis on-line que ajudam a cumprir os objectivos pretendidos [41] [42] [43].

4.3.2.2 Matplotlib

No detector digital, uma das informações mais importantes que pretendemos apresentar na interface gráfica são as amplitudes do sinal *copolar* e *crosspolar* e respectiva fase relativa. Designadamente a apresentação do historial mais recente (algumas horas) pode ser importante para o experimentador ter uma perspectiva imediata do que aconteceu num período de não vigilância. A consulta dos valores passados pode ser feita com o recurso a um traçado gráfico que apresenta o sinal ao longo do tempo.

No projecto em questão, iremos ter quatro traçados na interface gráfica. Estes correspondem à FFT do sinal e às já referidas amplitudes da componente *copolar* e *crosspolar* e à sua fase. Na construção inicial estas representações gráficas foram realizadas com as bibliotecas de *plot* incluídas no wxPython. Embora os resultados fossem satisfatórios, resolveu-se ensaiar aqui uma abordagem diferente, utilizando outras bibliotecas, e assim alcançar uma maior qualidade no traçado e um maior número de opções. Os novos traçados são agora baseados na biblioteca Matplotlib [44].

Matplotlib é uma biblioteca de *software* destinada a realizar traçados gráficos a duas dimensões. Este foi desenvolvido na linguagem Python e o seu princípio baseia-se na criação de gráficos de qualidade utilizando comandos simples. A ideia original partiu da tentativa de emular os comandos de *plot* incluídos no Matlab. Como já é sabido, os traçados gráficos produzidos pelo Matlab têm uma boa apresentação e são conseguidos com um número mínimo de comandos. Assim, o Matplotlib tem em vista o desenho de traçados de qualidade semelhante ao Matlab utilizando scripts em Python com um número de linhas bastante reduzido.

Vamos olhar para o código da Figura 4.9. Com apenas quatro instruções foi criada uma janela com o traçado gráfico de um vector com alguns valores. Foi ainda incluída automaticamente uma barra de ferramentas na parte inferior que permite não só interagir com o traçado, mas também a salvaguarda da figura num ficheiro de imagem. Esta é a utilização mais simples da biblioteca Matplotlib. É utilizado o módulo pyplot que

não só incluí comandos conhecidos do Matlab como produz gráficos com um nível de qualidade semelhante a este.

```
1: import matplotlib.pyplot as
   plt
2:
3: plt.plot([1,2,3,4])
4: plt.ylabel('some numbers')
5: plt.show()
```

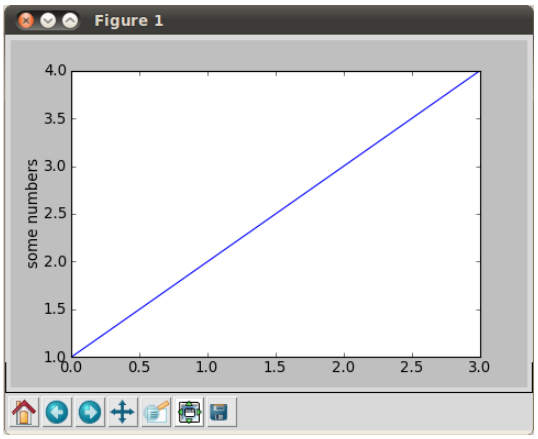


Figura 4.9: Exemplo de um gráfico realizado com o pyplot.

Embora esta solução sirva para algumas aplicações minimalistas, no nosso caso a utilização do módulo pyplot não pode ser considerada. A GUI do detector é construída sobre uma wx.Frame, e uma janela criada pelo pyplot não pode ser anexada à janela principal de forma directa.

O Matplotlib pode ser utilizado de três formas. A primeira foi descrita anteriormente e é através da importação do módulo pyplot. Uma segunda forma é utilizando o pylab. O pylab é instalado em conjunto com o Matplotlib e é uma junção entre o módulo pyplot e um outro chamado numpy. Isto resulta numa aproximação ainda maior ao Matlab, uma vez que são emuladas não só as funções de *plot* como também outras funções matemáticas. A utilização desta opção não é muito aconselhável, pois leva o utilizador a incorrer em más práticas que distorcem o verdadeiro funcionamento do Matplotlib. Por fim, uma outra forma reside na interface orientada a objectos.

Esta última forma de aplicar a biblioteca Matplotlib vai ser a solução para o problema do pyplot. Recorrendo à orientação a objectos irá ser possível anexar um gráfico produzido pelo Matplotlib à janela principal do detector digital criada em wxPython [45]. Apesar de ser a solução com a maior complexidade, esta permite ter um controlo total dos elementos do gráfico, o que possibilita moldar completamente a sua estrutura e assim obter os melhores resultados.

A integração do Matplotlib com o wxPython é possível graças à utilização de *backends* que fazem parte do Matplotlib. Um *backend* irá permitir ao Matplotlib interligar-se com outras interfaces, por exemplo outras bibliotecas, como é o caso das bibliotecas do wxPython. Enquanto o programador descreve as instruções para o traçado no código Python, o *backend* do Matplotlib trata de realizar o trabalho necessário para que a figura seja visível na janela estruturada pelo wxPython.

Para mostrar o funcionamento da ligação entre as referidas bibliotecas vamos recriar o gráfico da Figura 4.9 numa janela *top-level* do wxPython.

```
1: import wx
2: from matplotlib.figure import Figure
3: from matplotlib.backends.backend_wxagg import FigureCanvasWxAgg as
   FigureCanvas
```

```
4:
5: class MplFrame(wx.Frame):
6:  def __init__(self, title):
7:         wx.Frame.__init__(self, None, title=title, size=(300, 200))
8:
9:         self.figure = Figure()
10:               self.axes = self.figure.add_subplot(111)
11:               y = [1,2,3,4]
12:               self.axes.plot(y)
13:
14:               self.canvas = FigureCanvas(self,1, self.figure)
15:
16:               self.Show()
17:               self.Center()
18:
19: if __name__ == '__main__':
20:         app = wx.App()
21:         MplFrame('Matplotlib in Wx')
22:         app.MainLoop()
```

Muitas das instruções incluídas já são conhecidas dos exemplos mostrados na secção 4.3.2.1 relativa ao wxPython. Inicialmente são importados os módulos que serão utilizados. Além do já conhecido módulo wx, são agora importados dois componentes adicionais nas linhas 12 e 13. A "Figure" irá albergar uma ou mais instâncias do módulo "axes" para o traçado gráfico. O "FigureCanvasWxAgg" será o *container* da figura onde está desenhado o gráfico. Esta classe deriva do wx.Panel, logo pode ser incluída na janela principal como qualquer outro objecto. Neste caso é utilizado o *backend* WxAgg para permitir a ligação com o wxPython. Começando com o trabalho propriamente dito, na linha 5 é criada uma nova janela *top-level* que parte da classe wx.Frame. Após a sua inicialização nas linhas 6 e 7 é criada uma nova figura na linha 9. Para realizar o traçado é necessário criar uma instância de "axes" como é feito na linha 10. Os "axes" irão conter no seu interior os elementos básicos do gráfico como linhas ou texto. A linha 12 cria efectivamente o traçado dos pontos incluídos no vector *y* e na linha 14 é inicializado o *canvas* com o parâmetro *self* para indicar que o seu pai será a própria wx.Frame e com a advertência que este deverá acomodar a figura "self.figure". As últimas linhas, já explicadas num exemplo anterior, tratam de colocar o script wxPython em funcionamento.

Como resultado tem-se o traçado da Figura 4.10. Se comparado com o gráfico da Figura 4.9 verifica-se que de facto obtemos os mesmos resultados em termos do traçado. As diferenças que saltam à vista estão na falta da legenda do rótulo do eixo vertical e o desaparecimento da *toolbar* na parte inferior. Apesar de, estes elementos, não estarem incluídos neste exemplo não quer dizer que não possam existir. Aliás, esta é a principal vantagem da utilização do Matplotlib com interface orientada a objectos. Podemos adicionar, remover e editar os vários elementos do gráfico para cumprir com os requisitos de cada aplicação.

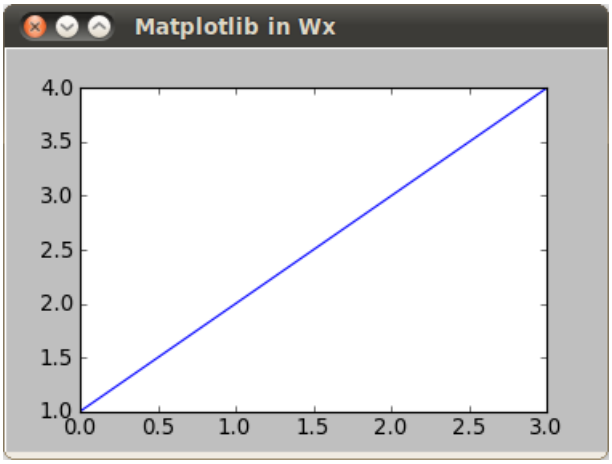


Figura 4.10: Exemplo gráfico de incorporação do Matplotlib no wxPython.

A descrição aqui feita permite tomar conhecimento das propriedades do Matplotlib e as suas possíveis aplicações. Os exemplos apresentados resumem-se a aplicações simples para facilitar a compreensão e não mostram todo o potencial desta biblioteca de *software*. No decorrer dos próximos capítulos irá ser apresentada a GUI do detector e com esta irão ser mostradas, em termos práticos, algumas das opções incluídas no Matplotlib que tornam esta ferramenta tão interessante.

5 Implementação do Sistema e Funcionamento do Detector

O capítulo 5 deste relatório aborda o *hardware* do *downconverter* e o *software* do detector digital. Irá ser feita uma descrição das modificações e das novas adições ao projecto inicial [15] [16].

Os conteúdos do presente capítulo irão ser divididos em duas partes, em que a primeira estará relacionada com o *hardware* do receptor e a segunda com o *software* desenvolvido para a detecção.

A secção relativa ao *hardware* estará focada na integração de novos filtros na unidade de acondicionamento de sinal, na revisão do *design* das placas de circuito impresso e também no *hardware* adicional desenvolvido para o teste global do sistema.

Na segunda parte do capítulo iremos proceder a uma explicação do funcionamento do *software* do detector, bem como das principais alterações e novas opções implementadas.

5.1 Hardware

5.1.1 Filtros

A filtragem do sinal num receptor é um dos pontos mais importantes na sua construção. A colocação de filtros ao longo da cadeia irá permitir eliminar, ou pelo menos atenuar, o ruído e componentes indesejáveis que poderão interferir com o sinal recebido.

Um filtro pode ter diversas naturezas. A sua construção pode ser feita, por exemplo, com base em componentes passivos, como resistências e condensadores, ou mesmo tirando partido das características piezoeléctricas dos cristais. No entanto, qualquer que seja a sua construção, o seu objectivo mantém-se inalterado e consiste na maior atenuação possível fora da largura de banda efectiva do filtro, tentando minimizar as perdas do sinal dentro da banda de passagem. Com isto será possível limitar a largura de banda do sinal ao longo da cadeia de *hardware*.

O *frontend* do receptor irá incluir dois filtros para frequências distintas, um de 2 GHz e outro de 10.7 MHz. A alteração dos filtros do projecto inicial tem em vista a adaptação do receptor às novas características conhecidas do sinal do *beacon* de satélite, bem como aumentar a robustez do mesmo relativamente a interferências externas.

5.1.1.1 Filtro para 2 GHz

O primeiro filtro a ser colocado na cadeia será destinado à filtragem da IF_1, de valor 2 GHz. Como já foi visto no capítulo 3, uma largura de banda de 100 MHz será suficiente para cumprir os requisitos, portanto este será um bom ponto de partida.

Como se pretende conseguir um circuito o mais imune ao ruído externo possível, decidiu-se substituir o primeiro filtro da cadeia. No projecto original, o filtro passa-banda centrado nos 2 GHz foi implemento com linhas *microstrip* acopladas que, apesar de

apresentar resultados aceitáveis, tinha umas perdas relativamente elevadas. Assim, uma nova solução foi encontrada ao recorrer a um filtro helicoidal.

Um filtro helicoidal é baseado num circuito ressonante realizado com um condutor interno em forma de hélice, rodeado por um escudo em forma circular ou rectangular altamente condutor. É possível realizar filtros deste tipo com factores de qualidade *Q* que chegam ao patamar dos milhares quando construídos para a gama VHF e UHF [46]. Estes filtros conseguem ser bastante selectivos e apresentam níveis baixos de atenuação na banda de passagem.

Figura 5.1: Exemplo de um circuito ressonante helicoidal e de uma solução comercial com 2 elementos ressonantes.

Apesar dos filtros helicoidais terem sido pensados para frequências mais baixas, existem fabricantes que produzem alguns modelos que incluem a frequência central e a largura de banda pretendida para o receptor. A Neosid [47] tem disponível uma gama de filtros com frequências centrais compreendidas entre os 816 e os 2450 MHz e com bandas de passagem que variam entre os 14 e os 80 MHz.

Um outro ponto importante na escolha é a resposta do filtro. Os filtros helicoidais podem ser constituídos por vários circuitos ressonantes e, com o aumento do número de elementos, menos *flat* será a resposta a sua resposta em frequência.

Figura 5.2: Variação da resposta em frequência de um filtro helicoidal com o número de elementos.

Optou-se por um filtro constituído apenas por dois circuitos ressonantes helicoidais, pelo que se espera que a sua resposta em frequência seja algo semelhante ao apresentado na Figura 5.2-b.

Segundo os dados do fabricante, o filtro escolhido tem uma frequência central sintonizável entre os 1900 e os 2000 MHz, com uma largura de banda de aproximadamente 45 MHz e uma atenuação na banda de passagem não superior a 2.5 dB. Estes valores vão de encontro àquilo que é desejado para o projecto.

Relativamente à montagem propriamente dita, este filtro revelou ser um pouco maior do que o esperado. Para além de uma maior imunidade a interferências provenientes do exterior, existia ainda a intenção de reduzir um pouco o espaço ocupado pelo filtro

inicialmente realizado com linhas acopladas. Devido ao tamanho que o este apresenta acabou por não ser produzida uma diferença significativa em termos de área ocupada pelo filtro.

5.1.1.2 Filtro para 10.7MHz

Enquanto o primeiro filtro, relativo à IF_1, pode possuir uma largura de banda larga, o filtro para a IF_2 tem de ser o mais estreito possível para que seja introduzida a menor quantidade de ruído no detector e evitar a saturação dos últimos amplificadores. No entanto, há algo aqui a ter em conta: o sinal do *beacon* de satélite irá sofrer variações de frequência devido a Doppler. Se estas variações forem muito significativas incorremos no risco do sinal cair fora da banda de passagem do filtro o que introduz dificuldades nos algoritmos de estimativa da NSD e da CNR do ponto de vista de aquisição do sinal e dificuldades no tratamento dos dados de propagação tornando mais difícil a estimativa da atenuação. Este problema tornou-se ainda maior com o conhecimento de que o sinal do *beacon* irá sofrer de efeito de Doppler e assim surgiu a necessidade de alargar o filtro da IF_2 para cumprir com os novos requisitos.

Tal como no primeiro projecto, vamos recorrer a um filtro a cristal monolítico. Devido às características piezoeléctricas dos cristais é possível criar circuitos ressonantes com um factor de qualidade *Q* elevado e assim obter filtros com uma frequência central bastante precisa e largura de banda bem mais estreita que filtros realizados com componentes passivos.

O filtro escolhido é o modelo 10M15A, construído pela Helpert, e está dimensionado para uma frequência central de 10.7 MHz, com uma banda de passagem de ±7,5 kHz e atenuação de 3 dB na referida banda. A sua montagem no *down converter* tem, no entanto, algo de diferente relativamente ao filtro da primeira IF. Ao contrário deste, o filtro de 10.7 MHz tem de "ver" à sua entrada e à sua saída uma impedância equivalente a 3000 Ω//2 pF e não 50 Ω. Com isto será necessário dimensionar um transformador de impedâncias para que seja possível a introdução do filtro no restante circuito.

O transformador de impedâncias que irá ser implementado terá como base um circuito ressonante LC com transformador capacitivo. Este circuito ressonante será dimensionado para um factor de qualidade baixo e posteriormente será dessintonizado para alcançar a impedância equivalente desejada.

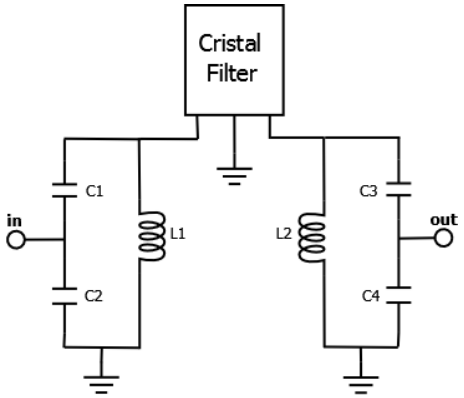


Figura 5.3: Malha de adaptação para filtro a cristal.

Uma vez que tínhamos disponíveis alguns componentes do projecto anterior, decidiu-se partir daqui para tentar encontrar uma solução.

Para realizar o transformador de impedâncias (Figura 5.3) é necessário encontrar os valores correctos para os condensadores e bobinas do circuito. Vamos dimensionar a malha de entrada e a malha de saída seguirá o mesmo raciocínio pois ambas terminam em 50 Ω. A metodologia para projectar estes circuitos pode encontrar-se em [48] e [49].

Um dos valores disponíveis para as indutâncias é 3.3 µH, valor este que é conseguido graças a um indutor variável da Burklin. Recorrendo à expressão (5.1) é possível determinar o condensador equivalente *C*. O seu valor será 67 pF.

$$C = \frac{1}{w_0^2 \,.\, L} \tag{5.1}$$

Com os valores de *C* e *L* é possível calcular a largura de banda *LB* e seguidamente o factor de qualidade *Q* do circuito ressonante recorrendo às expressões (5.2) e (5.3).

$$LB = \frac{1}{2\pi \,.\, C \,.\, R_T} \tag{5.2}$$

$$Q = \frac{f_0}{LB} \tag{5.3}$$

Chegamos então a uma estimativa de 880 kHz para a largura de banda *LB* que equivale a um factor de qualidade *Q* de 12.

Para o filtro a cristal é importante ter uma malha de adaptação de impedâncias com um factor de qualidade baixo. O valor de 12 estimado para o *Q* é um valor aceitável para esta construção, pelo que nos resta agora determinar os condensadores C_1 e C_2 da Figura 5.3.

Os condensadores C_1 e C_2 têm uma relação entre si que depende de uma relação de transformação *N*. Esta relação de transformação é definida pela seguinte expressão:

$$N = \left(\frac{R_T}{R_{I/O}}\right)^{1/2} \tag{5.4}$$

Com R_T igual a 2,7 kΩ e um $R_{I/O}$ de 50 Ω obtém-se para *N* um valor de 7,35.

$$C_1 = \frac{C_2}{N-1} \tag{5.5}$$

$$C_2 = N \,.\, C \tag{5.6}$$

Por fim, com recurso às expressões (5.5) e (5.6) chega-se a uma estimativa de 492 pF para o condensador C_2 e 78 pF para o condensador C_1.

5.1.2 Gerador de Ruído

Os geradores de sinal de laboratório são pensados para produzirem sinais de teste com o menor ruído possível. Em situações em que se pretenda simular um sinal real com um elevado patamar de ruído será necessário introduzi-lo posteriormente utilizando um gerador de ruído. Não estando disponível um gerador de ruído para 2 GHz e ainda porque eram necessárias duas fontes de ruído descorrelacionadas optou-se por desenvolver uma fonte de ruído "artesanal" sem recorrer a possíveis díodos geradores de ruído. Um gerador de ruído não só produzirá o ruído adicional requerido mas também terá um circuito que permite adicionar ao ruído o sinal de teste.

Nas estimativas realizadas no capítulo 3 vimos que após a primeira conversão de frequência do sinal (IF_1) a potência do sinal seria cerca de -110.55 dBW. Vimos também que o valor máximo da CNR esperado irá rondar os 55 dB. Isto resulta numa potência de ruído *N* de valor -165.55 dBW.

Para gerar o ruído pretendido iremos aproveitar as características não ideais dos amplificadores. Todos os componentes que compõem um sistema RF produzem ruído que é adicionado ao sistema e os amplificadores não são excepção. O que irá ser feito na prática será a amplificação do ruído de uma cadeia de amplificadores.

A quantidade de ganho que será necessária irá depender do ruído introduzido pelos amplificadores e em especial do primeiro. Uma possível solução está representada na Figura 5.4. Os amplificadores utilizados serão MMIC's da Mini-Circuits.

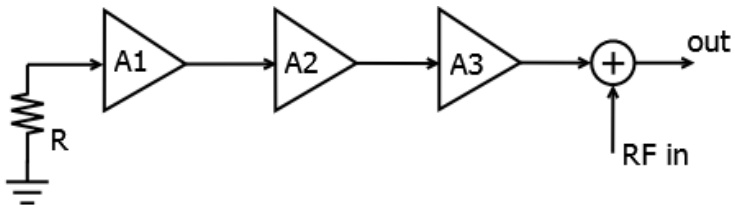


Figura 5.4: Diagrama do gerador de ruído.

A densidade espectral do ruído à saída do amplificador A_3 irá ser dada pela expressão:

$$N_o = k \,.\, T_{eq} \,.\, G_{Total} \tag{5.7}$$

em que T_{eq} é a temperatura equivalente do ruído à entrada do sistema e G_{total} o ganho total fornecido pelos amplificadores.

O primeiro amplificador da cadeia será um Gali-55+. Este amplificador apresenta uma figura de ruído de 3.3 dB e será este valor que irá determinar a temperatura equivalente do sistema.

$$T_{eq} = T_0 + \left(10^{N_F/10} - 1\right) . T_0 \quad (5.8)$$

A temperatura equivalente à entrada do circuito T_{eq} será 620 *K*, assumindo que a temperatura de referência T_0 terá o valor 290 *K*.

Sabendo que N_o irá corresponder à densidade espectral de ruído estimada à entrada da cadeia de *hardware* do receptor (-165.55 dBW) e com o valor de T_{eq} determinado anteriormente podemos agora recorrer à expressão (5.7) para calcular o ganho necessário. O ganho total G_{total} será 35 dB.

Para adicionar os 35 dB de ganho serão utilizados três amplificadores. Tanto o amplificador A_1 como o A_2 são Gali-55+ e fornecem um ganho de 18.5 dB a 2 GHz. O amplificador A_3 será um Gali-21+ capaz de um ganho de 13.1 dB para a mesma frequência de 2 GHz. Em teoria estes amplificadores fornecem em conjunto um ganho de 50.1 dB. Como se pode verificar, este valor está um pouco acima da estimativa realizada anteriormente. O desenho do gerador de ruído inclui um atenuador fixo no final da cadeia de amplificadores para permitir o ajuste do nível de ruído. O ganho adicional acaba por ser propositado pois irá aumentar a margem para o referido ajuste.

Em termos de funcionamento, o ruído introduzido pelos amplificadores irá ser amplificado pelos mesmos e posteriormente será somado ao sinal RF por um *combiner* SCN-2-22 também da Mini-Circuits. O sinal resultante será o equivalente possível ao sinal proveniente do *beacon* de satélite que irá ser introduzido pelo *hardware* do *frontend* comercial que se pretende adquirir. A entrada do gerador de ruído irá estar terminada com uma carga adaptada de 50 Ω mas pode ser substituída por uma resistência.

De notar que como irão existir duas cadeias de *hardware* no *frontend* analógico (sinal *copolar* e sinal *crosspolar*) também serão construídos dois geradores de ruído, um para cada componente do sinal.

5.1.3 *Layout* das Placas do Circuito

A revisão do *hardware* do projecto inicial tem em vista não só a adaptação do sistema às novas características do *beacon* de satélite mas também a melhoria da unidade de IF dos 2 GHz e preparação da unidade para ser encerrada numa caixa para se tornar mais imune a interferências externas.

Começando pela placa de acondicionamento de sinal (Figura 5.5), foram realizadas modificações para suportar os novos filtros: na parte relativa à IF_1 (2 GHz) o filtro de linhas acopladas foi substituído pelo filtro helicoidal e após a conversão para a segunda IF (10.7 MHz) existem apenas as linhas necessárias para a soldagem de um único filtro a cristal que será o filtro da IF_2. Foram introduzidos planos de massa em zonas não utilizadas do PCB para imunizar ainda mais o sistema a interferências e realizado um redimensionamento geral para reduzir o seu tamanho com vista a um futuro

encapsulamento. Todos os planos de massa na face superior estão ligados à face inferior por abundantes furos metalizados.

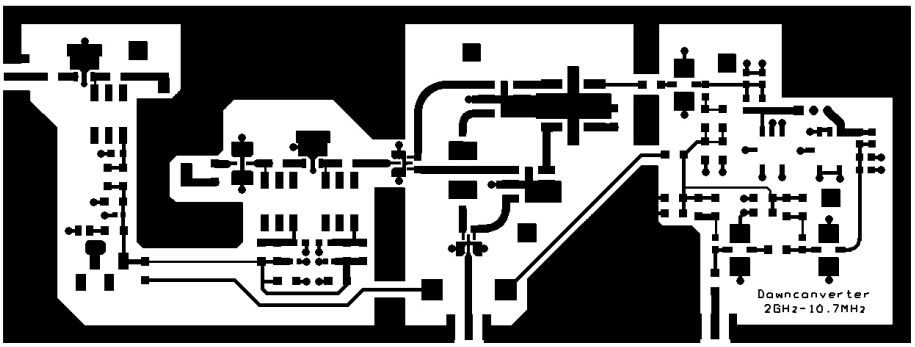


Figura 5.5: PCB da unidade de acondicionamento de sinal (*down converter*).

Uma vez que a unidade de síntese apresenta um bom desempenho desde o seu projecto inicial não foram realizadas modificações significativas em termos de componentes. Procedeu-se apenas à substituição dos reguladores de tensão e à remoção da parte que implementaria uma DDS (*Direct Digital Synthesis*) por não ser utilizada. Tal como a unidade anterior também foi feito um redimensionamento para minimizar o tamanho da placa de circuito impresso.

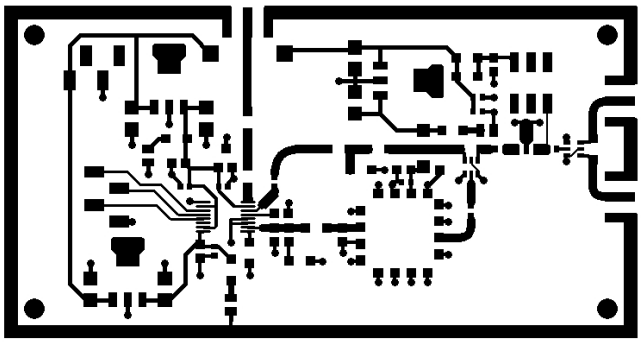


Figura 5.6: PCB da unidade de síntese (oscilador local).

Por último, a Figura 5.7 mostra o *layout* do PCB do gerador de ruído. Esta nova placa foi adicionada ao projecto para a realização de testes ao sistema. Um ponto relevante que se destaca neste design é que, apesar de serem utilizados alguns dos mesmo amplificadores do *down converter*, o *choke* de RF que realiza alimentação dos MMIC's não utiliza um componente dedicado. Como neste caso não é necessário um circuito ideal optou-se por usar bobinas construídas manualmente evitando o uso dos mais dispendiosos *chokes* RF.

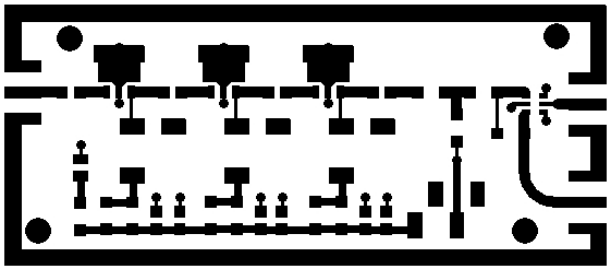
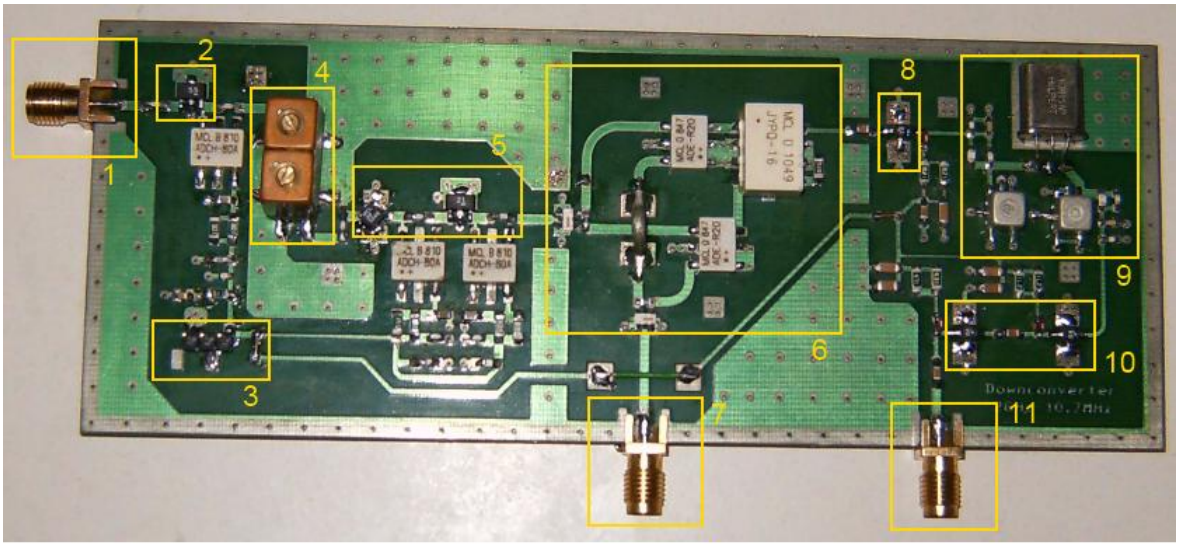


Figura 5.7: PCB do gerador de ruído.

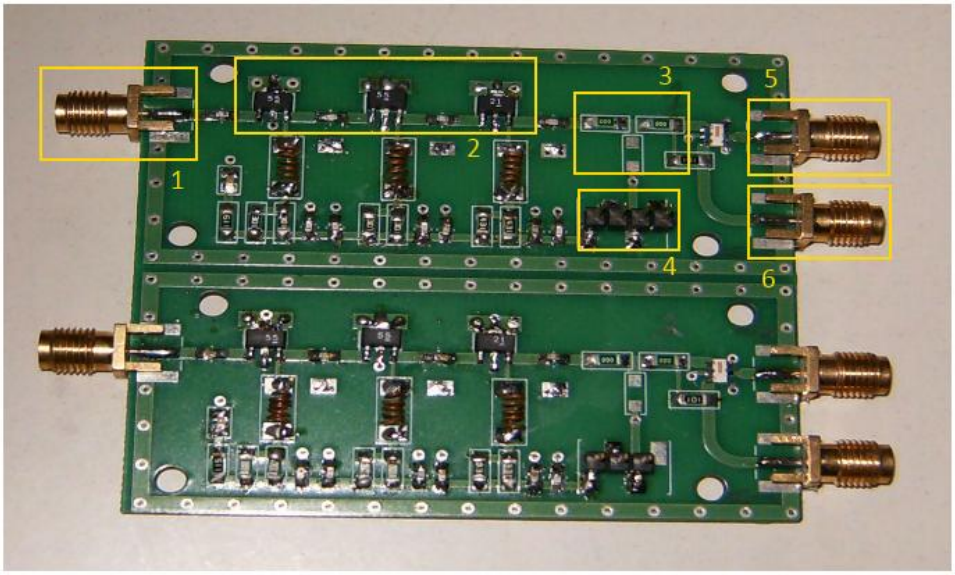
5.1.4 Montagem das Placas de *Hardware*

Após a produção das novas placas do receptor procedeu-se à soldagem dos componentes. As figuras desta secção mostram o novo *hardware* construído incluindo uma enumeração das partes mais importantes.



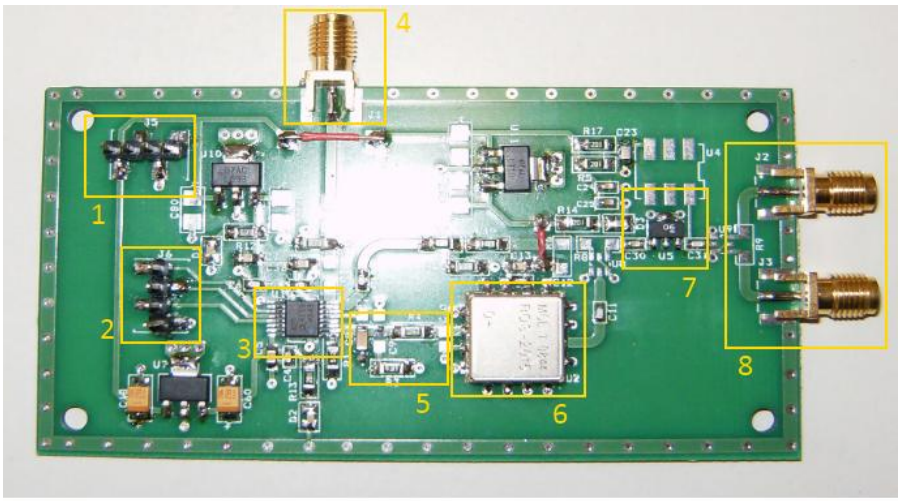
1. Conector para o sinal de entrada;
2. Amplificador (2 GHz);
3. Conector para alimentação;
4. Filtro Helicoidal (2 GHz);
5. Amplificadores (2GHz);
6. Filtro de rejeição de imagem;
7. Conector de entrada do oscilador local;
8. Amplificador (10.7 MHz);
9. Filtro a cristal e circuito ressonante (10.7 MHz);
10. Amplificadores (10.7MHz);
11. Conector para o sinal de saída

Figura 5.8: Placa de acondicionamento do sinal.



1. Conector para sinal de entrada (terminado com carga de 50 ohm por defeito);
2. Cadeia de Amplificadores (2 GHz);
3. Atenuador fixo;
4. Conector de alimentação;
5. Conector para sinal de saída;
6. Conector para sinal de entrada (gerador de sinal);

Figura 5.9: Placa do gerador de ruído.



1. Conector para alimentação;
2. Conector para programação do chip PLL;
3. Chip PLL;
4. Conector para oscilador de referência;
5. Filtro de malha;
6. VCO;
7. Amplificador (2 GHz);
8. Conectores do sinal de saída;

Figura 5.10: Placa da unidade de síntese.

5.2 Software

5.2.1 Estrutura do *Software* do Detector

Os sinais do *beacon* de satélite na IF de 10.7 MHz serão introduzidos no kit de rádio digital USRP. A partir deste ponto todas as acções sobre o sinal irão ser realizadas por *software* utilizando o GNURadio. O USRP irá realizar uma última *down convertion* para a banda base a partir da qual, após análise espectral, serão determinados os vários parâmetros dos sinais.

Antes de se entrar em detalhe sobre o *software* do detector digital será importante conhecer a sua estrutura e funcionamento.

O *software* pode ser dividido em três partes principais: a interface gráfica (GUI), o processo de *tuning* e o processo de aquisição. Para uma melhor descrição fazemos referência à Figura 5.11.

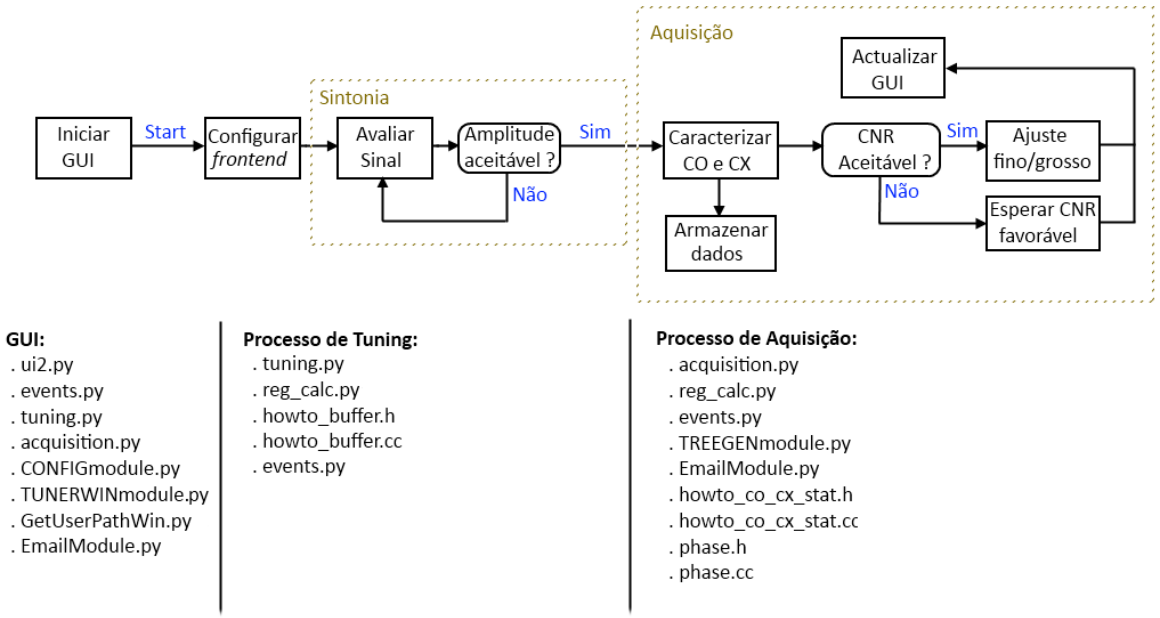


Figura 5.11: Estrutura do software do detector e módulos associados.

O primeiro passo será a inicialização da interface de utilizador. Esta irá permitir configurar os vários parâmetros do sistema antes de começar o trabalho útil bem como dar início a todo o processo.

Após configuradas as opções desejadas dá-se início ao processo de *tuning*. Este processo intermédio tem em vista a validação do sinal antes da aquisição. Sucintamente, após ter sido configurado o oscilador local do *frontend* para a frequência desejada (normalmente 2.0107 GHz) é avaliada a amplitude do sinal por análise espectral. Se for atingido o factor de qualidade desejado (amplitude estável) o sistema salta para o passo seguinte. Caso contrário o sistema mantém-se num *loop* contínuo até ocorrerem melhores condições. A última frequência do NCO usada no processo de *tuning* irá ser utilizada como frequência central no passo seguinte. Durante este processo o operador pode monitorizar o espectro que é sucessivamente analisado.

Terminado o processo de *tuning* o sistema passa para o processo de aquisição. Partindo da frequência central descoberta no processo de *tuning* são realizadas estimativas para as características do sinal que posteriormente são armazenadas em ficheiros Matlab. Com uma análise espectral é, de seguida, avaliada a necessidade de um ajuste de frequência. Se for necessário será realizado um ajuste no oscilador local (ajuste grosso) ou no NCO (ajuste fino). Caso ocorra uma situação de baixa CNR o sistema não realiza qualquer modificação de frequência mas continua a fazer a aquisição dos dados pesquisando a risca numa largura de banda estreita de forma a não ser iludido por picos de ruído especialmente quando a CNR é muito baixa. Quando for atingido o valor de CNR aceitável o sistema retoma o funcionamento normal. O processo de aquisição repete-se continuamente até o utilizador o interromper.

5.2.2 A Interface do Utilizador

Enquanto o GNU Radio realiza os cálculos e estimativas do sinal é necessária uma forma de visualizar resultados em tempo real bem como algo que permita a configuração do sistema de forma simples e directa. Isto é conseguido graças a uma janela gráfica desenvolvida exclusivamente para o efeito.

Os principais requisitos da interface do utilizador são os seguintes:

- Visualização dos dados relativos à amplitude do sinal *copolar*, *crosspolar* e fase relativa em forma de traçado gráfico em tempo real num período de tempo longo;
- Apresentação de estimativas instantâneas das características do sinal;
- Representação gráfica do espectro de frequência do sinal *copolar*;
- Possibilidade de configurar os parâmetros do sistema através de ficheiros configuração.

Para que todo o sistema funcione correctamente, a GUI terá de funcionar em paralelo com o processamento digital de sinal e não deverá utilizar excessivamente os recursos do sistema.

O *software* do detector irá utilizar programação concorrente para permitir o funcionamento simultâneo das várias partes do *software* e comunicação entre elas. Com o detector em funcionamento as várias partes do programa irão ser executadas como *threads*. Para a actualização dos valores da interface um evento será despoletado que irá permitir aceder aos vectores de dados para retirar os valores mais recentes. Isto irá impedir que duas *threads* acedam à mesma posição de memória ao mesmo tempo. Depois de conseguidos os novos dados será necessário o envio dos mesmos para a thread da GUI e posterior actualização dos vários elementos da interface. Estes objectivos são conseguidos graças a eventos construídos para o efeito presentes no módulo *events.py*.

Como foi referido, um evento irá permitir a actualização da interface sem interromper em definitivo as outras threads do *software*. Um cuidado que deverá ser tido é o esforço necessário para realizar o traçado dos gráficos. À medida que os dados vão sendo recolhidos mais e mais pontos serão adicionados ao traçado e como todo o gráfico é

desenhado a cada actualização um maior esforço computacional será necessário a cada adição de um novo valor. Como a maior prioridade terá de ser sempre dada ao processamento digital de sinal terá de ser encontrado um compromisso para a actualização da interface do utilizador.

A GUI do detector sofreu uma renovação geral desde a sua versão original. A nova abordagem surgiu na tentativa de reparar uma falha que existia na primeira versão do *software* mas posteriormente acabou por dar origem a algo mais elaborado. As bibliotecas adicionais introduziram novas possibilidades que resultam numa melhor iteração com o utilizador.

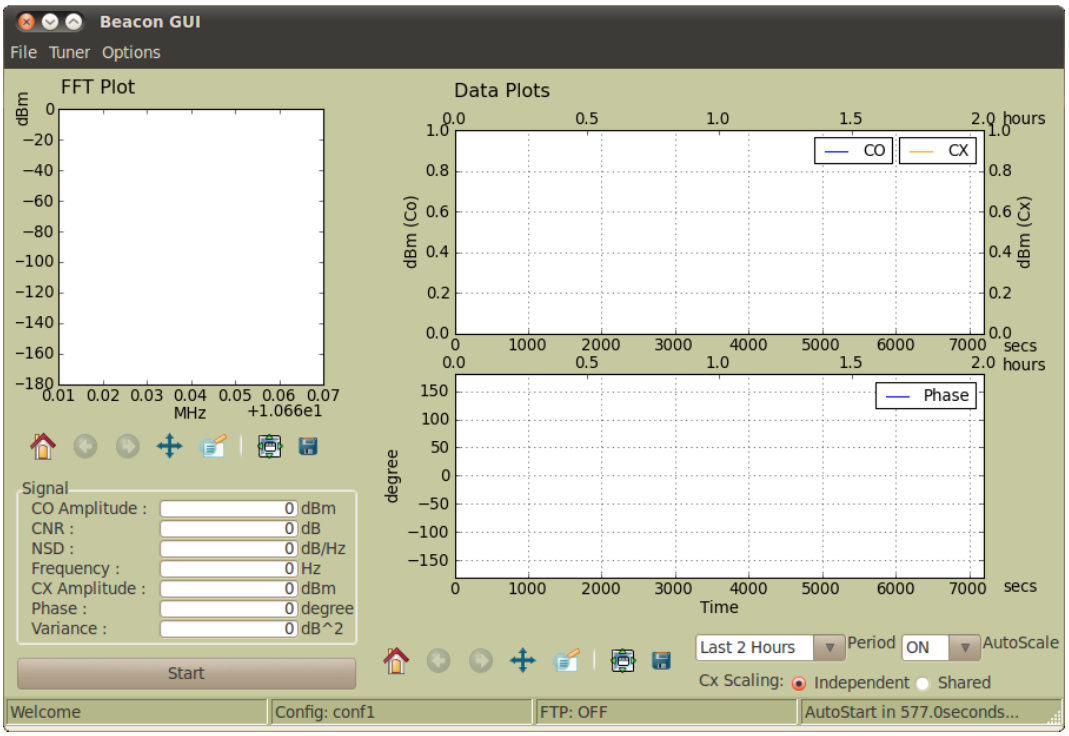


Figura 5.12: Janela principal da GUI.

Na Figura 5.12 é mostrada a janela principal da nova GUI. A estrutura da janela, bem como a maior parte das opções, são realizadas utilizado as funcionalidades do wxPython. A principal diferença encontra-se na forma como são traçados os gráficos. É aqui que entra a biblioteca Matplotlib, que não só produz melhores traçados como introduz opções adicionais. O código para a criação desta interface encontra-se no módulo *ui2.py*.

A janela principal é constituída por três áreas para traçado, uma lista para as estimativas das características do sinal e várias opções e menus para configuração.

Nas áreas para os gráficos, à direita, serão desenhados os traçados de amplitude do sinal CO e CX na área superior e da fase relativa na área inferior. Estes gráficos irão ter um histórico até 10 horas, 1 amostra tirada a cada 2 segundos, o que resulta num total de 18000 amostras em cada um destes traçados. Estes valores são suficientes para tomar conhecimento do passado recente com resolução temporal suficiente mantendo a estabilidade do detector em termos de esforço computacional. Este conjunto de gráficos tem disponível um conjunto de opções dedicado que actua sobre as escalas dos mesmos (Figura 5.13). Poderão ser seleccionadas várias escalas para visualização entre 15 minutos

e 10 horas existindo a possibilidade de desactivar o *auto-scaling* de ambos os gráficos. Isto poderá ser útil no caso de se pretender verificar variações numa escala mais apertada, impedindo o escalonamento automático. Uma última opção permite, nos traçados de amplitude do sinal CO e CX, separar ou tornar independentes as escalas dos gráficos. Caso a opção "Independent" esteja activa, a escala de amplitude do sinal CO estará na linha vertical à esquerda e a do sinal CX estará no lado oposto.

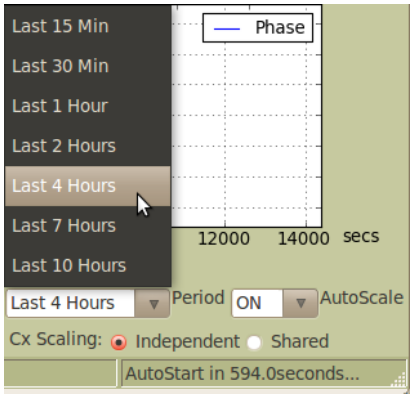


Figura 5.13: Opções de escala (CO, CX, Phase).

A última área de traçado gráfico serve para mostrar o espectro de frequência do sinal *copolar* em tempo real. Embora seja dado o valor instantâneo da amplitude o acompanhamento das variações num traçado gráfico é mais evidente, podendo ainda ser verificado se de facto as estimativas são realizadas sobre o sinal ou sobre uma qualquer espúria que indevidamente entrou no sistema. Tal como os outros gráficos, uma barra de ferramentas está presente para facilitar o estudo dos traçados (Figura 5.14). Esta barra de ferramentas está incluída no pacote Matplotlib e pode ser ligada a qualquer gráfico produzido por esta biblioteca. São incluídas funções de *zoom*, deslocamento da área do gráfico e mesmo a possibilidade de gravar um instantâneo do traçado num ficheiro de imagem.

Figura 5.14: Matplotlib - Barra de ferramentas por defeito.

Conhecidos os novos gráficos vamos agora passar para a configuração dos parâmetros do sistema. A versão original da GUI já continha opções para modificar parâmetros mas apesar do módulo existir já se encontrava desactualizado perante novas necessidades de parametrização e exigia uma integração completa e mais funcional no *software* e designadamente na interface com o utilizador. Desta forma o módulo *CONFIGmodule.py* foi totalmente reestruturado e as modificações permitem agora carregar as variáveis a partir de ficheiros de configuração e utilizá-las no processamento.

Para aceder à janela de edição dos parâmetros do sistema selecciona-se no menu a opção "File" e de seguida "Edit Configuration". Esta janela pode ser vista na Figura 5.15. Existem ainda outras duas opções sob este menu que permitem carregar um ficheiro de configuração e visualizar a última configuração carregada.

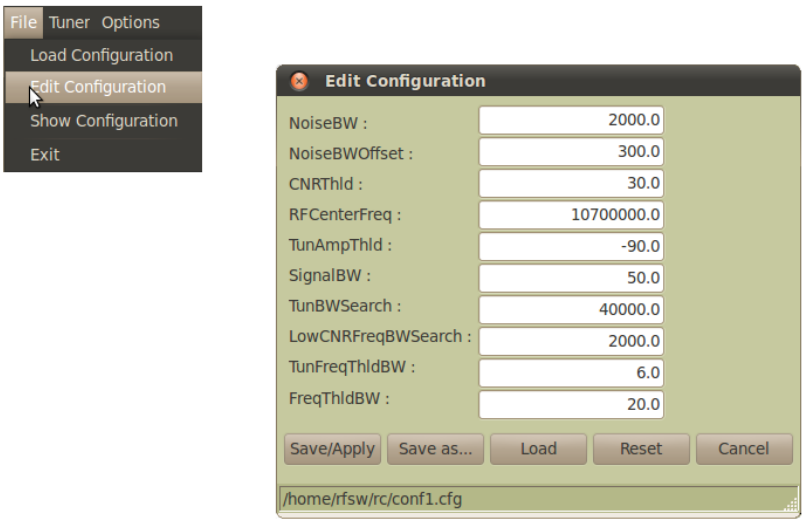


Figura 5.15: Janela de edição de ficheiros de configuração.

Na janela de edição existem cinco opções disponíveis. É ainda incluída uma barra de estado na parte inferior que permite confirmar qual o ficheiro de configuração em utilização.

O botão "Save/Apply" guarda as modificações realizadas ao actual ficheiro de configuração e aplica-as ao sistema. A opção "Save as…" realiza uma operação semelhante ao "Save/Apply" com a particularidade de dar a possibilidade de guardar os parâmetros num novo ficheiro de configuração. O "Load" carrega um ficheiro na interface para edição, o "Cancel" anula todas as alterações efectuadas e fecha a janela e por fim o "Reset" carrega os valores por defeito na interface. Quando a GUI é inicializada é criado um ficheiro de configuração por defeito caso este não exista. A janela de configuração está impedida de efectuar qualquer alteração sobre a configuração por defeito. O *software* irá guardar sempre a última configuração utilizada.

Todos os parâmetros da lista são utilizados em alguma parte do programa:

- NoiseBW: Largura de banda (acima e abaixo da frequência central) que será utilizada para a estimativa da potência do ruído – Valor por defeito: 2000 Hz;
- NoiseBWOffset: Distância, em Hz, da frequência central a partir da qual o espectro é usado na estimativa da NSD. Este *offset* será a margem superior e inferior relativamente frequência central uma vez que serão utilizados *bins* de ambas as partes do espectro para o cálculo da potência do ruído – Valor por defeito: 300 Hz;
- CNRThld: Valor de CNR mínimo aceitável para o processo de aquisição. Abaixo deste valor não será feito qualquer ajuste de frequência. – Valor por defeito: 30 dB;
- RFCenterFreq: Frequência central utilizada como referência (frequência inicial do NCO) – Valor por defeito: 10700000 Hz;
- TunAmpThld: Amplitude mínima do sinal que, no processo de *tuning*, será necessária para identificar um sinal válido e iniciar o processo de aquisição – Valor por defeito: -20 dBm;

- SignalBW: Largura de banda considerada para a estimativa da potência do sinal em condições normais – Valor por defeito: 50 Hz;
- TunBWSearch: Largura de banda onde será pesquisado sinal *copolar* no processo de *tuning* com vista a calcular a frequência de partida do NCO antes de iniciar a aquisição – Valor por defeito: 40000 Hz;
- LowCNRFreqBWSearch: Largura de banda para a localizar o sinal no espectro caso o valor da CNR esteja abaixo do valor mínimo – Valor por defeito: 1000 Hz;
- TunFreqThldBW: Variação de frequência máxima aceitável sem que ocorra um ajuste de frequência no processo de *tuning* – Valor por defeito: 10 Hz (±5 Hz da última frequência estimada);
- FreqThldBW: Variação de frequência máxima aceitável sem que ocorra um ajuste de frequência no processo de aquisição – Valor por defeito: 10Hz (±5 Hz da última frequência estimada);
- FreqDesvMax: Máxima frequência de desvio do sinal antes de o sintetizador ser actualizado – Valor por defeito 3 kHz.

Para além da edição dos ficheiros de configuração na janela apresentada, a modificação dos valores poderá ser realizada directamente nos ficheiros. Ao recorrer à segunda opção há que ter o cuidado de não modificar a ordem dos vários elementos. A troca da ordem dos parâmetros irá resultar num mau funcionamento do programa.

Durante o processo de aquisição irão ser guardados ficheiros de dados com diversas informações. Uma das novas adições ao *software* do detector é a possibilidade realizar cópias de segurança para uma pasta que será partilhada por outros computadores utilizando o serviço Dropbox [50]. No final de cada dia serão copiados os dados recolhidos que correspondem apenas ao mês corrente. Desta forma é evitada a cópia de um grande volume de dados. Ao aceder à opção "Options"→"Path Configuration" irá surgir uma janela (Figura 5.16) que permite configurar a pasta de destino das cópias de segurança. Esta função poderá também ser desactivada. Esta é uma facilidade interessante pois pode perder-se um tempo significativo na gestão manual dos ficheiros.

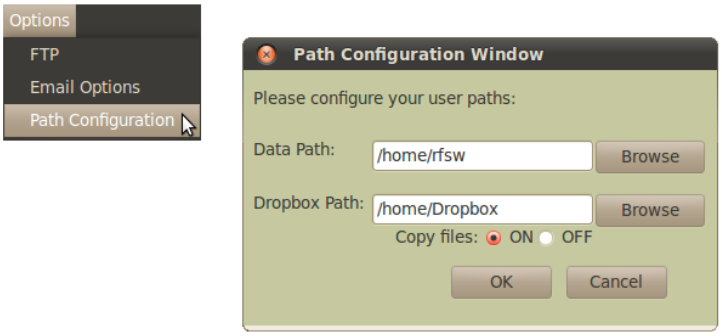


Figura 5.16: Janela de configuração dos *paths* do programa.

Na mesma janela existe ainda um campo para configurar o "Data Path". Este *path* corresponde à localização pretendida para o armazenamento dos ficheiros criados. Embora a realização de cópias de segurança não seja obrigatória o campo

correspondente ao “Data Path” deverá ser sempre preenchido. Na primeira vez que o *software* do detector for iniciado uma janela semelhante à mostrada na Figura 5.16 irá surgir. O programa não irá prosseguir se não forem fornecidos os endereços destes directórios. Estas funções estão construídas no módulo *GetUserPathWin.py*.

Uma outra nova função desenvolvida é a possibilidade de alertar o utilizador caso o sinal seja recebido em más condições (*EmailModule.py*). Quando ocorrer um caso de baixa CNR um aviso será enviado para o endereço de correio electrónico do utilizador que lhe permite tomar conhecimento do sucedido. Posteriormente o utilizador poderá verificar se existe algum problema no receptor ou se se trata apenas de um caso de condições climatéricas adversas que provocaram uma atenuação temporária do sinal. Este aviso será enviado no máximo uma vez em cada 30 minutos. Para uma fácil configuração dos dados de acesso ao *email* existe a opção “Email Options” sob o menu “Options” que mostra a janela da Figura 5.17. São suportados os servidores de *email* do Gmail e da Universidade de Aveiro.

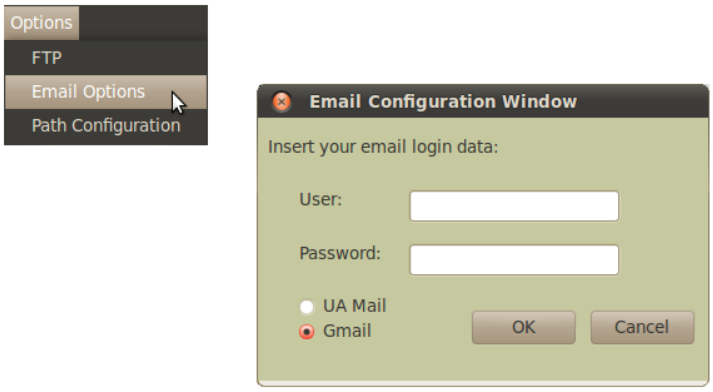


Figura 5.17: Janela de configuração dos dados de acesso ao endereço de *email*.

Ainda no menu “Options” existe uma outra opção denominada “FTP”. Esta opção provém da versão original do programa e foi ideia do seu autor para a cópia dos ficheiros de dados para servidores de FTP. Como foi introduzida uma nova abordagem para a realização de cópias de segurança esta função não sofreu qualquer revisão ou teste uma vez que não será utilizada.

Como se pode reparar, o menu principal da GUI inclui uma opção chamada “Tuner”. Esta opção permite mostrar uma janela secundária relativa ao processo de *tuning* que contém um conjunto de estimativas do sinal recebido e uma representação gráfica do espectro de frequências. O traçado do espectro de frequências é igualmente desenhado com a biblioteca Matplotlib. Ao ser iniciado o processo de *tuning* a janela associada irá ser mostrada automaticamente e manter-se-á activa enquanto este processo decorrer.

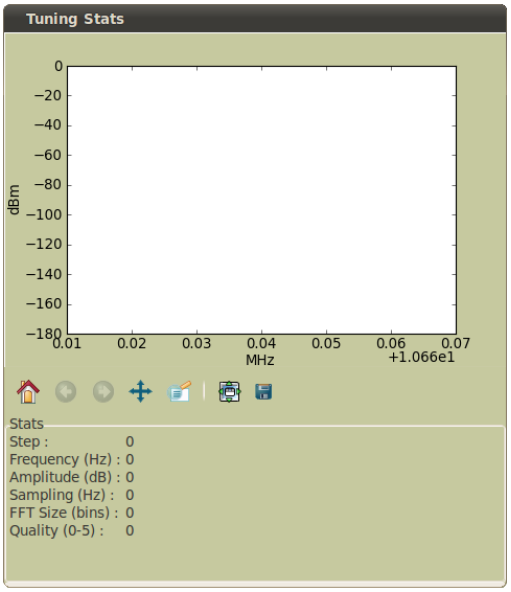


Figura 5.18: Janela do processo de *tuning*.

Finalmente, o início da recepção poderá ser realizado de duas formas: premindo o botão "Start" na parte inferior esquerda da GUI ou de forma automática caso exista um período de inactividade superior a 10 minutos. A segunda hipótese será importante no caso de um corte de energia. O computador poderá estar configurado para se ligar automaticamente quando for restaurada a energia eléctrica e, adicionando o programa do detector aos processos de arranque, o programa será inicializado de forma automática começando de imediato a recolha de dados sem que seja necessária a intervenção do utilizador.

5.2.3 Processo de *Tuning*

Como já foi explicado é importante garantir que o sinal é recebido em boas condições antes de começar a recolha de dados. O processo de *tuning* irá fazer uma análise inicial ao sinal e permitir uma configuração óptima do receptor antes do trabalho útil. Será realizada uma avaliação baseada em estimativas de potência e de frequência e posterior sintonia, caso se justifique, e assim conseguir um seguimento do sinal.

O módulo principal neste processo é o *tuning.py*, onde é definido o *flow graph* e as funções de controlo associadas. Teremos então duas classes principais: a classe "TunerFlowGraph" e a classe "tuner".

5.2.3.1 Diagrama de blocos do Processo de *Tuning*

A classe "TunerFlowGraph" implementa o diagrama de blocos do processo de *tuning*. A sua estrutura mantém-se intacta desde a sua primeira versão pelo que vamos apenas proceder à sua análise. Na Figura 5.19 temos um esquemático do referido diagrama.

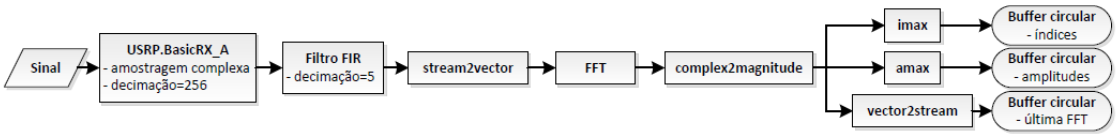


Figura 5.19: *Flow graph* do processo de *tuning* [15].

À entrada do detector irá estar um sinal proveniente do *frontend* analógico de frequência 10.7 MHz. Pretende-se analisar apenas 50 kHz de largura de banda pelo que será necessário realizar uma decimação. Como a amostragem é realizada a 64 MS/s e o factor máximo de decimação suportado pela FPGA é 256 será necessário realizar a decimação em dois passos. A FPGA reduzirá a largura de banda num factor de 256 e seguidamente um filtro FIR passa-baixo tratará da restante decimação com um factor de 5 que resultará nos 50 kHz pretendidos. A frequência de corte do filtro será suficientemente larga para não excluir informação útil (f_{corte}>7 kHz). As amostras são quantidades complexas pelas razões já mencionadas em 4.1.2.1.
Neste ponto também será definida a frequência do NCO para a conversão para a banda base. Pretende-se um sinal junto a DC pelo que a frequência central será 10.7 MHz.

Depois de adquirido um vector com as amostras do sinal é feita uma FFT. É necessária uma resolução fina o suficiente para uma estimativa fiável da frequência central do sinal, por tal a FFT terá 32768 amostras o que corresponde a uma resolução de cerca de 1,5 Hz.

Agora com o sinal digitalizado e traduzido para o domínio das frequências resta determinar as suas características. Depois de converter os sinais complexos para o seu equivalente em magnitude, os blocos “imax” e “amax” derivados das bibliotecas do GNU Radio determinam, respectivamente, o índice do *bin* de maior amplitude e a magnitude desse índice. Com estes valores irão ser estimadas a amplitude e a frequência central do sinal.

Os últimos blocos do diagrama dizem respeito a *buffers* de dados. Estes blocos foram criados de raiz em C++ no projecto original (*howto_buffer.cc/.h*) e permitem manter um histórico das amostras do sinal e das estimativas calculadas para ser possível a apresentação dos gráficos e a actualização dos restantes campos da GUI.

5.2.3.2 Funcionamento

A classe “tuner” fará o controlo do processo de *tuning*. Aqui serão tomadas as acções para a avaliação do sinal e seu seguimento até serem reunidas as condições para o início da última fase do programa do detector.

Após carregadas as variáveis do ficheiro de configuração e inicializadas as restantes dá-se início ao *flow graph* explicado anteriormente. O primeiro passo será a programação do sintetizador do oscilador local, ou seja, a configuração do *frontend*. Enquanto o *software* original simplesmente programava o sintetizador para se obter 2.0107GHz no oscilador local, a nova versão faz um varrimento numa largura de banda configurável para achar a frequência mais adequada. Desta forma serão tomados os seguintes passos:

1. Programar NCO para a frequência central 10.7MHz;

2. Programar *chip* do sintetizador para 2.0107GHz menos metade da largura de banda de pesquisa;
3. Determinar amplitude do sinal e registar em vectores os valores da amplitude e frequência correspondente do oscilador local;
4. Dar um salto de frequência de 5 kHz e repetir o passo 3 até varrida a largura de banda pretendida;
5. Percorrer o vector de amplitudes e determinar o índice de valor máximo de amplitude e a frequência a que corresponde esse índice;
6. Programar o sintetizador do oscilador local para a frequência descoberta no ponto anterior;

A programação do sintetizador será feita com recurso ao protocolo de comunicação SPI escrevendo nos registos do *chip* a configuração desejada. As funções para a programação do sintetizador encontram-se no módulo *reg_calc.py*. O salto de frequência de 5 kHz corresponde ao *channel spacing* na síntese de frequências.

Uma vez configurado o *frontend* o programa segue como o original. São retirados os últimos cinco valores dos *buffers* de amplitude e frequência e realizadas médias. É feita uma avaliação do sinal verificando se a média de amplitude está ou não acima do mínimo imposto. Se não estiver o factor de qualidade é posto a zero. Se estiver soma-se 1 ao factor de qualidade e verifica-se se existe um desvio de frequência. Caso a frequência esteja fora da largura de banda imposta (± 5 Hz por defeito) procede-se a uma sintonia fazendo um ajuste no NCO. Os buffers serão também configurados com a nova frequência central.

Esta sequência repete-se continuamente até ser atingido um factor de qualidade de valor 5. Quando isto acontecer o processo de *tuning* termina forçando o início do processo de aquisição com a última frequência do NCO. Este processo pode ser relevante especialmente nos casos em que o receptor é ligado pois o OCXO demora alguns minutos a atingir a frequência final e durante este período a frequência da IF apresenta uma deriva significativa.

5.2.4 Processo de Aquisição

O processo de aquisição é a parte mais importante do programa do detector. Este processo irá estimar e avaliar as características do sinal, fazer o seu seguimento e registar os valores recolhidos em ficheiros de dados binários para futura análise.

Tal como o módulo de *tuning*, o script que implementa o processo de aquisição (*acquisition.py*) incorpora duas importantes classes que definem a estrutura e o controlo do processo. São estas a classe “AcquisitionFlowGraph” e a classe “acquire”. As modificações realizadas irão ser descritas ao longo desta secção.

5.2.4.1 Diagrama do Processo de Aquisição

Comecemos por perceber o diagrama de blocos do processo de aquisição construído na classe "AcquisitionFlowGraph". Este diagrama encontra-se na Figura 5.20.

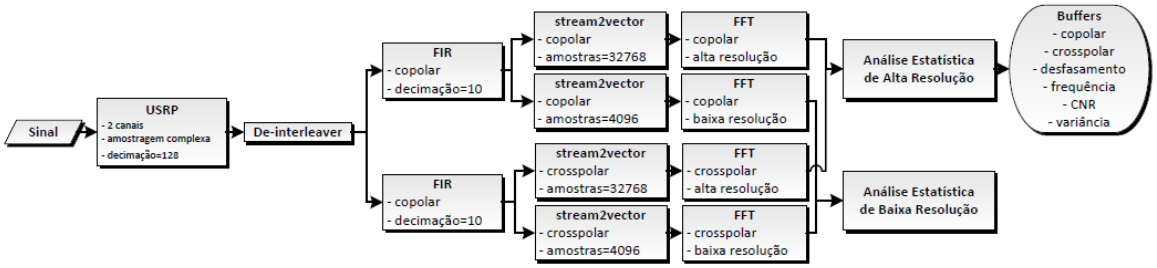


Figura 5.20: *Flow graph* do processo de aquisição [15].

Como pretendemos também analisar a componente despolarizada do sinal, o USRP terá agora de lidar com dois sinais de entrada complexos: o sinal *copolar* e o sinal *crosspolar*. Tal como no processo de *tuning* é considerado que uma largura de banda de 50 kHz será suficiente para o estudo do sinal e da mesma forma terá de ser realizada uma decimação. Uma vez que o factor de decimação da FPGA fica reduzido a metade (128) com a utilização de dois canais, o factor de decimação do filtro FIR passa-baixo terá de ser aumentado para o dobro (10), daí os valores mostrados nos blocos do diagrama. O bloco "De-interleaver" tratará de separar os *streams* de dados dos sinais *copolar* e *crosspolar* uma vez que estes virão multiplexados num único *stream* quando enviados pela interface USB. A amostragem será mais uma vez complexa pois, como já foi referido, irá permitir a distinção entre as frequências negativas e as frequências positivas facilitando o ajuste de frequência no seguimento do sinal além de obviamente permitir a detecção do *crosspolar* em amplitude e fase relativa.

O passo seguinte será a FFT, mas neste caso teremos uma situação diferente. Um dos fenómenos que se pretende estudar é a cintilação, ou seja, variações rápidas (até 2 a 3 Hz em contraste com os restantes com 0.01 Hz ou menos) de amplitude do sinal. Uma FFT como uma elevada resolução não é sensível a estas variações mas será sempre necessária para a estimação da frequência central do sinal com precisão. Desta forma serão realizadas duas FFT's distintas: uma FFT mais lenta mas de elevada resolução com 32768 amostras (≈1.5 Hz) e uma segunda FFT rápida com 8 vezes menos resolução de 4096 amostras (≈12 Hz). A FFT rápida, de período mais curto, permitirá ver as flutuações que o sinal terá ao longo do tempo.

Os últimos blocos dizem respeito à análise estatística e aos buffers de dados. As principais modificações ao projecto original são realizadas aqui. Estes blocos foram desenvolvidos para estimar as características do sinal, escreverem em ficheiros e manterem um histórico de dados, no entanto algum trabalho teve de ser despendido para rectificar e optimizar os cálculos.

5.2.4.2 Blocos de Análise Estatística e Buffers de Dados

Os ficheiros *howto_co_cx_stat.cc/.h* e as dependências *phase.cc/.h* irão implementar os blocos de análise estatística e, na nova versão, também os *buffers* de dados.

As estimativas do sinal serão conseguidas com a implementação em linguagem C++ dos algoritmos apresentados no subcapítulo 4.2. A forma como são determinadas a potência do *copolar*, a CNR, a NSD, a variância e a frequência mantém-se como a original mas com algumas correcções para melhorar os resultados. Todos estes algoritmos, exceptuando a variância, dependem directamente da frequência central, ou seja do *bin* que a representa. O sinal irá estar espalhado numa largura de banda (50 Hz) e por tal achar o *bin* de maior amplitude e converter o seu índice para a frequência respectiva não é uma boa prática. Isto foi corrigido no código original com a implementação efectiva da expressão (4.2) onde são usadas as riscas mais importantes do sinal para determinar a sua frequência com uma resolução superior à da FFT.

O código para as restantes estimativas (amplitude do CX e fase relativa) foi construído na íntegra por utilizar uma nova fórmula de cálculo. Mantendo a mesma estrutura em termos de entradas e saídas procedeu-se à renovação do código que irá agora implementar os algoritmos explicados em 4.2.3:

```
for (unsigned i = 0; i < VectorCo.size(); i++)
{
        Sum += real(VectorCo.at(i)*conj(VectorCo.at(i)));
        CoCxSum += VectorCx.at(i)*conj(VectorCo.at(i));
}

*CxPhase = atan2(imag(CoCxSum),real(CoCxSum))*180/pi;
if (*CxPhase < -180) *CxPhase = *CxPhase + 360;
if (*CxPhase > 180) *CxPhase = *CxPhase - 360;
*CoAmp = sqrt(Sum);
*CxAmp = abs(CoCxSum)/(*CoAmp);
```

Esta função encontra-se no *script phase.cc* e irá devolver a amplitude do sinal *copolar*, a amplitude do sinal *crosspolar* e a fase relativa entre os dois sinais correlacionados. Como entradas estarão os vectores complexos que contêm o espectro das riscas mais significativas dos sinais CO e CX.

Uma outra adição é a limitação da banda na procura do *bin* de maior potência em caso de baixa CNR. O *software* já inclui um controlo de largura de banda para a estimativa de potência do sinal (quando CNR < 40 dB) que diminui esta banda de 50 Hz para 12 Hz para não incluir potência de ruído indevidamente e ter resultados mais precisos. No entanto, a procura do *bin* de maior potência quando a CNR atinge o valor mínimo (CNR < 30 dB) continuava a ser feito em todo o espectro do sinal. Com esta revisão, enquanto a CNR se encontre acima do limite aceitável é feito um varrimento em todo o espectro. Se a CNR se degradar até abaixo do limite a procura é feita numa banda limitada para que o sistema não se agarre a alguma espúria ou pico de ruído e acabe por derivar para fora da largura de banda do filtro FIR.

Neste bloco existem diversas variáveis que serão configuráveis pelo utilizador antes do início da aquisição. Foram introduzidos métodos para que estas variáveis sejam

carregadas do ficheiro de configuração na inicialização do bloco. Caso não seja possível, por alguma razão, o carregamento das variáveis serão tomados valores por defeito.

Relativamente aos *buffers* de dados, a razão para a geração de *buffers* neste mesmo bloco deve-se a um problema que os *buffers* do projecto original revelaram: decorrido um período aleatório mas nunca superior a 3 horas o programa interrompia o funcionamento. Uma vez que o erro está associado a gestão de memória e não a uma excepção produzida por uma instrução mal formulada, tentar apontar a origem da falha prometia ser um processo difícil e longo e por tal foi utilizada uma nova abordagem. Em vez de *buffers* configuráveis num bloco à parte, como os utilizados no processo de *tuning*, foram definidos *buffers* no bloco análise estatística com tamanhos reduzidos (3 valores) à excepção do relativo à FFT que terá o tamanho igual ao número de amostras da mesma. Enquanto isto será o suficiente para retirar amostras instantâneas das estimativas do sinal para actualização dos parâmetros da GUI, um problema se levanta quanto ao histórico de amostras necessário ao traçado das amplitudes dos sinais CO e CX e sua fase relativa. Isto é resolvido no código Python na classe "acquire". Valores retirados dos *buffers* de amplitude e fase irão ser adicionados de forma contínua em vectores criados em Python. Um novo valor irá substituir o mais antigo, o que na prática resultará num deslizamento nos gráficos. Irá ser mantido um histórico de 10 horas, uma amostra em cada 2 segundos, como já foi explicado durante a apresentação da nova interface do utilizador.

5.2.4.2.1 Ficheiros de Dados das Estimativas do Sinal

O bloco produzido pelos ficheiros *howto_co_cx_stat.cc/.h* inclui ainda o código necessário para a criação e escrita de ficheiros de dados binários. Os novos ficheiros de dados terão a particularidade de poderem ser carregados de imediato no Matlab e assim serem analisados directamente.

Um ficheiro binário Matlab (.mat) é constituído por um cabeçalho e pelos dados que se pretendem armazenar. Actualmente existem várias versões de ficheiros MAT que foram sendo modificadas à medida que o Matlab sofreu actualizações. Apesar de serem necessárias bibliotecas próprias para a criação destes ficheiros existe a possibilidade de gerar ficheiros MAT de mais baixo nível utilizando, por exemplo, a linguagem C ou C++ [51]. O Matlab apresenta suporte para as versões mais antigas pelo que não haverá problemas com os ficheiros criados desta forma.

O primeiro passo será a escrita do cabeçalho. Este cabeçalho terá diversas opções que deverão ser escritas numa determinada ordem e deverão corresponder exactamente aos conteúdos do ficheiro. Caso isto não aconteça o ficheiro será ilegível. A estrutura do cabeçalho será a seguinte:

```
typedef struct {
        long type;
        long mrows;
        long ncols;
        long imagf;
        long namelen;
} MATLAB_MATRIX_HEADER;
```

O parâmetro “type” irá conter diversas informações. Este poderá ser escrito na forma de um número inteiro com 4 dígitos segundo o formato MOPT, em que cada letra representa um dígito. O número que será escrito no nosso caso em particular será 0000. O dígito relativo ao M diz respeito à arquitectura utilizada pela máquina. O valor 0 irá corresponder a IEEE *Little Endian*. O O será sempre 0, este é um algarismo reservado para futuras opções. O P será o formato em que os dados serão armazenados e será 0 porque iremos escrever *floats* com precisão dupla (64-bit). Por último o T, também de valor 0, irá definir o tipo de matriz que será totalmente numérica. Cada valor escrito irá ocupar 8 bytes com esta configuração. Nos tempos que correm os discos rígidos têm grandes dimensões e baixo custo pelo que o tamanho de 8 bytes não deverá causar problemas apesar do elevado volume de dados.

Os parâmetros seguintes definem, pela ordem que são apresentados, o número linhas e o número de colunas da matriz de dados, a *flag* assinalando números complexos e o tamanho do nome da matriz. A nossa matriz terá sempre 9 linhas e o número de colunas será actualizado automaticamente à medida que sejam escritos novos dados. Só serão armazenados valores reais pelo que a *flag* relativa à existência de parte imaginária deverá estar a 0.

A escrita no ficheiro binário terá de ser realizada pela seguinte ordem: escrever cabeçalho; escrever nome da matriz; escrever dados. Isto resultará num ficheiro MAT suportado pelo Matlab.

Resta agora saber o que efectivamente será escrito nos ficheiros e para tal olhemos para a Tabela 5.1. Nesta tabela está feita uma representação da estrutura da matriz de um ficheiro de dados bem como o seu conteúdo. À medida que novos valores são escritos no ficheiro novas colunas serão adicionadas. O conteúdo de cada coluna é óbvio excepto o das duas últimas linhas: “NCO Update Flag” e “Timestamp”.

O “NCO Update Flag” é um campo para verificar o processo de sintonia. Cada vez que o NCO é reprogramado este parâmetro é colocado a 1 e desta forma podemos ver com facilidade quando houve um ajuste de frequência.

Tabela 5.1: Estrutura dos dados nos ficheiros binários.

Sample	**Next Sample**
CO Amplitude (dBm)	CO Amplitude (dBm)
CX Amplitude (dBm)	CX Amplitude (dBm)
Phase (degree)	Phase (degree)
CNR (dB)	CNR (dB)
Frequency (Hz)	Frequency (Hz)
Variance (dB^2)	Variance (dB^2)
NSD (dB/Hz)	NSD (dB/Hz)
NCO Update Flag	NCO Update Flag
Timestamp (s)	Timestamp (s)

O outro campo desconhecido, o “Timestamp”, foi adicionado para ter uma referência temporal das estimativas. Uma vez que estamos a armazenar dados em binário o “Timestamp” utilizado será o tempo em segundos em formato do *UNIX* aquando a escrita

de uma nova coluna. É possível transformar o valor deste campo numa data completa no Matlab utilizando os seguintes passos:

1. Determinar o tempo no formato Excel (Windows): $Excel_{Time} = UNIX_{Time}/86400 + 25569$, em que 86400 corresponde ao número de segundos de um dia e 25569 o número de dias entre 1970 e 1900 (diferença de anos das referências temporais do Windows e do UNIX);
2. Converter o formato de data Excel para o formato de data numérico do Matlab com o comando *x2mdate*;
3. Tornar o número representativo da data do Matlab numa *string* legível com a função *datestr*.

Uma última nota é a forma como o número de colunas é actualizado. Uma vez que o cabeçalho é escrito de uma só vez este terá de voltar a ser escrito com o novo número de colunas uma e outra vez até à criação de um novo ficheiro ou interrupção do programa. No caso de interrupção forçada surgiu a dúvida de que poderia haver uma corrupção do ficheiro se o número de colunas fosse incrementado e esta não fosse escrita na sua totalidade. Isto de facto acontece. Para solucionar este problema o número de colunas só é incrementado após a escrita de todos os valores da coluna. Desta forma, em caso de interrupção forçada, a coluna incompleta, apesar de ser escrita, não é mostrada na leitura do ficheiro e já não ocorre a corrupção do mesmo.

Igualmente parecem redundantes algumas das variáveis armazenadas. É sempre preferível armazenar toda a informação que descreve o funcionamento interno para, em caso de necessidade, diagnosticar anomalias.

5.2.4.3 Funcionamento

O funcionamento do processo de aquisição pode ser explicado com a descrição da classe “acquire”.

O primeiro passo será, tal como no processo de *tuning*, o carregamento das variáveis a partir do ficheiro de configuração e a inicialização de outras que sejam necessárias. De seguida são criados os directórios, caso não existam, para o armazenamento dos ficheiros de dados com auxílio do módulo *TREEGENmodule.py*. Feito isto é iniciado o *flow graph* do processo com a frequência central determinada no processo de tuning.

O procedimento seguinte será a aquisição dos dados. Nos blocos de análise estatística serão calculadas as estimativas do sinal que posteriormente serão escritas de forma contínua em ficheiros .mat.

Com as amostras da aquisição dos dados será avaliado o sinal recebido. Ao contrário do processo de *tuning* agora são feitas estimativas com algoritmos próprios, pelo que o limite aceitável para efectuar o seguimento irá ser dado pela CNR. O valor da CNR, determinado pela média das últimas 3 estimativas instantâneas, irá ser comparado ao valor de CNR mínimo. Caso esteja abaixo deste valor não é efectuado qualquer ajuste de frequência mas o sistema prossegue com a escrita dos ficheiros e actualização da GUI. Neste caso o processo fica a aguardar melhores condições e enviará um aviso via correio

electrónico ao utilizador para o alertar do caso de baixa CNR. Se pelo contrário a CNR tiver um valor acima do limite procede-se à análise e ajuste de frequência.

O ajuste de frequência irá permitir fazer o seguimento à custa da frequência do NCO. É sabido que a frequência do sinal irá ter variações diárias não superiores a 5 kHz. Um ajuste de frequência fino com a reprogramação do NCO evitará que o sinal caia fora da largura de banda do filtro FIR de decimação devido ao deslizamento de frequência. Este ajuste de fino irá ser realizado quando existirem variações significativas (±5 Hz por defeito) da última frequência estimada. Além da variação de frequência diária, com o decorrer do tempo, os desvios de frequência do *beacon* e dos osciladores locais do receptor poderão causar um desvio acumulado sistemático de alguns kHz por ano. Isto poderá implicar a necessidade de um ajuste esporádico no oscilador local para impedir que o sinal fique além da frequência de corte do filtro da IF_2 de 10.7 MHz (±7.5 kHz) no *frontend* analógico. Assim, se a frequência estimada estiver acima ou abaixo 3 kHz dos 10.7MHz da frequência central do filtro a cristal, irá ser feita a reprogramação do sintetizador ±5 kHz (*channel spacing* do sintetizador de frequências), dependendo do sentido do desvio, e feito um ajuste no NCO de ± 2kHz para compensar o *offset* introduzido. Estima-se que 3 kHz será um valor aceitável para se proceder à reconfiguração do *frontend*. Após ser realizado qualquer ajuste de frequência no NCO a nova frequência central será configurada nos blocos de estimativas e *buffers* de dados.

A sequência aquisição, avaliação e seguimento irá ser realizada de forma continuada até o programa ser interrompido. A avaliação e o seguimento irão repetir-se sequencialmente a cada 2 segundos. A cada actualização as novas estimativas irão ser passadas à GUI com recurso às funções do módulo *events.py* e posteriormente serão actualizados os elementos da interface.

Os ficheiros de dados para as duas FFT's irão ser fechados a cada 6 horas e ao final de cada dia independentemente de não terem sido concluídas as 6 horas de aquisição num ficheiro. Também ao final do dia, se a opção estiver activa, irá ser feita uma cópia de segurança dos dados para uma pasta de partilha Dropbox

6 Teste e Avaliação do Sistema

Neste capítulo irão ser descritos os testes realizados para a validação do *hardware* e *software* do receptor de propagação e feita uma análise ao seu desempenho.

Iremos começar por testes individuais às várias partes do sistema e depois partir para provas com a montagem completa do sistema.

6.1 Testes Individuais

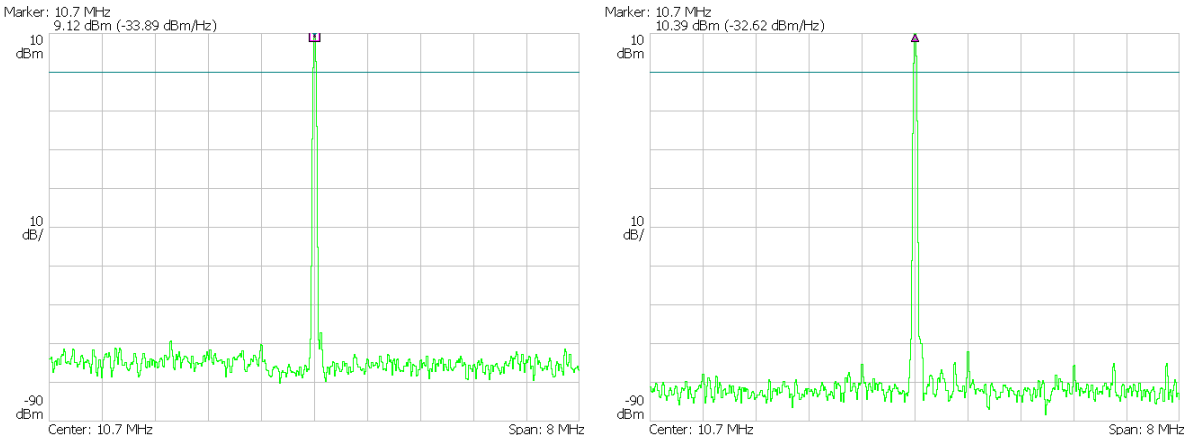
O receptor de propagação desenvolvido no âmbito deste trabalho é um conjunto de *hardware* e *software* que permite medir a amplitude de dois sinais e a sua fase relativa. Antes de procedermos à junção das várias partes é necessário garantir que cada uma delas se encontra a funcionar devidamente e cumpre os requisitos para os quais foi desenvolvida.

6.1.1 Placas de Acondicionamento de Sinal e Conversão de Frequência

Ao manterem uma estrutura semelhante à do projecto original é esperado que as novas placas de acondicionamento de sinal tenham também um desempenho idêntico. No entanto a introdução de novos elementos, como diferentes filtros ou mesmo o redimensionamento global, exige um novo conjunto de testes. São testadas as duas placas de acondicionamento de sinal: uma para o sinal *copolar* e outra para o sinal *crosspolar*.

Para o teste de cada uma necessitamos de dois sinais de entrada: um que corresponderá ao *beacon* e outro ao oscilador local, pelo que serão utilizados dois geradores de sinal numa primeira fase. Vamos começar por verificar qual o ganho efectivo de cada uma das IF's e ao mesmo tempo a conversão de frequência da IF_1 para a IF_2.

No início da cadeia irá estar um sinal com uma frequência de 2 GHz (IF_1) e -80 dBm de amplitude e na entrada do LO teremos um sinal de 2.0107 GHz com 10 dBm. Na saída irá surgir um sinal com uma frequência que será a diferença entre 2.0107 GHz e 2 GHz que corresponderá à IF_2 (10.7 MHz). Este sinal deverá um ganho de 90 dB.



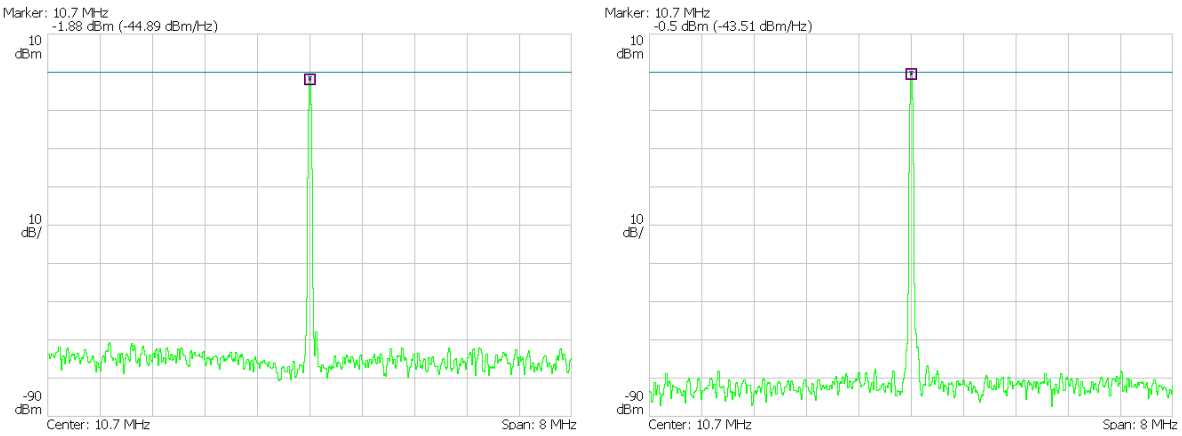
a) Down converter 1 b) Down converter 2

Figura 6.1: Espectro de um sinal de teste após ganho e conversão de frequência.

O espectro do sinal de saída de cada uma das cadeias encontra-se na Figura 6.1. O sinal está centrado em 10.7 MHz em ambos os canais provando o correcto funcionamento dos *down converters*. Quanto ao ganho, o *down converter* 1 apresenta um ganho aproximado de 90 dB e o *down converter* 2 um ganho de cerca de 91.5 dB. Ambas as unidades cumprem os objectivos em termos de ganho e apresentam resultados muito próximos (cerca de 1.5 dB de diferença). Recomenda-se o uso da cadeia de maior ganho para o *crosspolar* que é um sinal consideravelmente mais fraco.

Um outro aspecto importante é a rejeição da frequência imagem durante a conversão de frequências. Como já foi discutido anteriormente, o ruído da frequência imagem irá degradar o sinal da IF_2. Será então essencial saber o quanto a frequência imagem é atenuada.

Para simular a frequência imagem irá ser introduzido um sinal de teste com a mesma amplitude que na primeira situação (- 80dBm), mas agora este irá ter uma frequência de 2.0214 GHz onde deverá estar a frequência imagem do sinal do *beacon*. Os espectros dos sinais de saída para cada uma das placas de acondicionamento e *down convertion* encontram-se na Figura 6.2.



a) Down converter 1 b) Down converter 2

Figura 6.2: Espectro da frequência imagem após ganho e conversão de frequência.

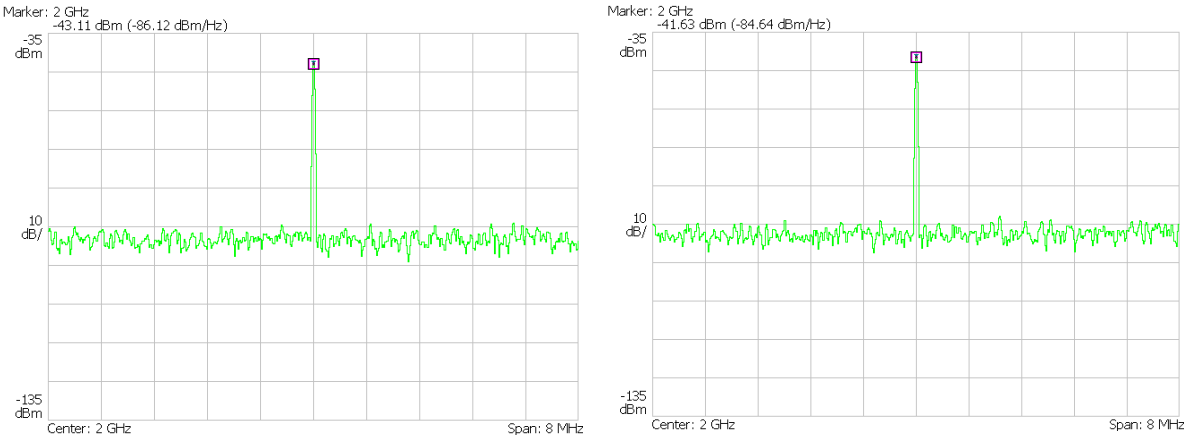
Em ambas as unidades a rejeição da frequência imagem ronda os 11 dB. Isto corresponde à adição de uma potência de ruído de 0.079 ao sinal, o que resulta numa degradação da CNR em 0.33 dB. Este valor é suficientemente baixo contudo poderá merecer uma optimização no futuro pois valores de cerca de 16 dB eram o objectivo.

6.1.2 Gerador de Ruído

Não sendo parte integrante do receptor, o gerador de ruído não deixa de ser um elemento importante. Uma vez que o satélite Alphasat ainda não se encontra em órbita a única forma de produzir um sinal com a CNR esperada é somando um patamar de ruído ao sinal produzido artificialmente por um gerador de sinais de laboratório.

Iremos ter dois geradores de ruído: um para simular o ruído do sinal *copolar* e outro para simular ruído do sinal *crosspolar* que proporcionam a necessária descorrelação.

O primeiro ponto que irá ser avaliado é o ganho efectivo de cada uma das cadeias de *hardware*.



a) Gerador de ruído 1; b) Gerador de ruído 2;

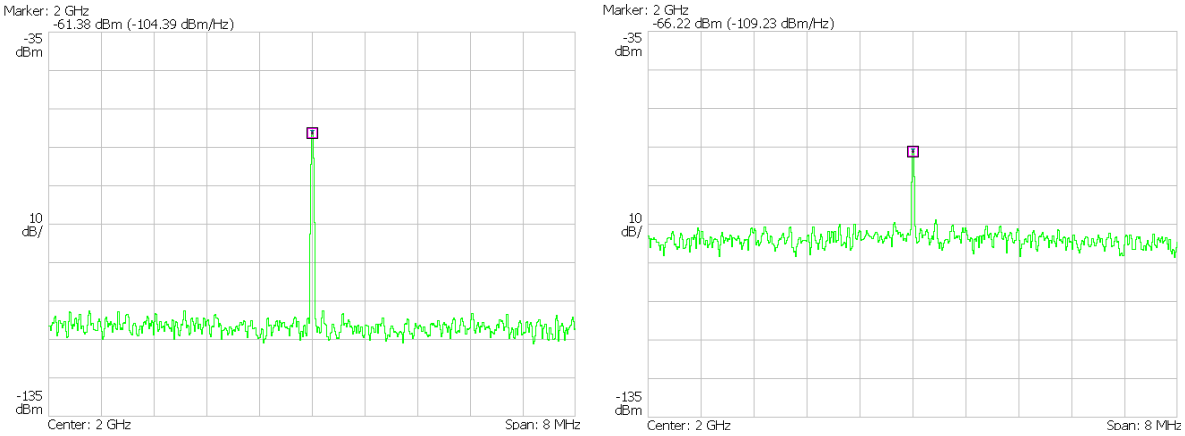
Figura 6.3: Espectros para estimativa de ganho dos geradores de ruído.

Como referência iremos colocar na entrada dos geradores de ruído um sinal com frequência 2 GHz de amplitude -80 dBm.

O espectro do sinal amplificado para cada um dos geradores de ruído encontra-se na Figura 6.3. O ganho para o sinal de referência a 2GHz para o primeiro gerador é de 38 dB e para o segundo 40 dB. De acordo com as estimativas feitas em 5.1.2 é necessário um ganho mínimo de 35 dB para o aumento do nível de ruído, pelo que o ganho obtido pelos dois geradores se encontra dentro do pretendido.

Apesar dos valores serem aceitáveis não podemos deixar de notar que são perdidos mais de 10 dB em cada canal da fonte de ruído relativamente ao valor teórico esperado. Cerca de 3.5 dB são perdas no combinador de saída portanto faltam ainda 6.5 dB. Estas perdas deverão estar relacionadas com a utilização dos *chokes* artesanais em vez de *chokes* RF. A impedância introduzida pelas bobinas não deverá ser o suficiente elevada para minimizar as fugas do sinal para o circuito de alimentação pelo que é perdida parte da sua potência.

Os geradores de ruído irão ainda fazer a soma do ruído produzido pelos mesmos com um sinal proveniente de um gerador de sinal. O próximo teste irá verificar a boa operação dos combinadores bem como a subida do patamar de ruído. Para ainda ser visível depois do combinador usou-se um sinal com -60 dBm.



a) Sinal de referência (2GHz) b) Sinal de referência + ruído

Figura 6.4: Gerador de ruído 1 - sinal + ruído.

Como é visível nos espectros da Figura 6.4 o patamar de ruído é aumentado ficando mais próximo do sinal de referência, ou seja, é conseguido um aumento da NSD.

Este processo de combinação acaba ainda por introduzir uma pequena atenuação no sinal de teste devido ao combinador. Isto pode ser corrigido com o ajuste posterior da amplitude do sinal no gerador de sinais.

O segundo gerador de ruído apresenta resultados semelhantes ao primeiro apenas com a diferença de ganho que foi vista no primeiro teste.

Temos portanto um *hardware* que permite gerar dois sinais coerentes com ruído AWGN descorrelacionado e ao mesmo tempo gerar uma CNR arbitrária pelo controlo da potência de sinal do gerador. A amplitude do sinal pode, posteriormente à adição de ruído, ser variada pela introdução de um atenuador (a CNR não variará). O gerador de ruído mostrou alguma tendência a oscilar com uma carga reactiva na saída contudo a introdução do atenuador (já previsto) estabilizou a unidade.

6.1.3 Detector Digital

Várias alterações ao *software* do detector foram realizadas para corrigir erros, afinar algoritmos e introduzir novas funcionalidades. Antes de se proceder ao teste conjunto com o *frontend* analógico será essencial assegurar a estabilidade deste *software*.

A montagem utilizada para teste será simples: um sinal proveniente de um gerador de sinais com uma frequência de 10.7MHz (IF_2) e amplitude -10 dBm será introduzido no USRP. Iremos simular apenas o funcionamento do detector utilizando um canal do USRP, ou seja, apenas o estudo com o sinal que representará o *copolar*. O outro canal e os algoritmos de cálculo associados irão ser testados na montagem completa do receptor.

Antes de se iniciar o processo de aquisição foram testadas a abertura, edição e fecho dos ficheiros de configuração bem como as opções adicionais de configuração de endereço de *e-mail* e directórios de escrita de dados. Todos apresentam os resultados esperados.

Uma das principais modificações na interface do utilizador é a nova forma de desenhar os gráficos utilizando a biblioteca Matplotlib. Nada melhor para provar o seu correcto funcionamento do que recorrer às opções introduzidas por esta biblioteca para salvar um instantâneo de um gráfico. Na Figura 6.5 encontra-se o espectro de frequências do sinal de teste ainda sem o ruído aditivo.

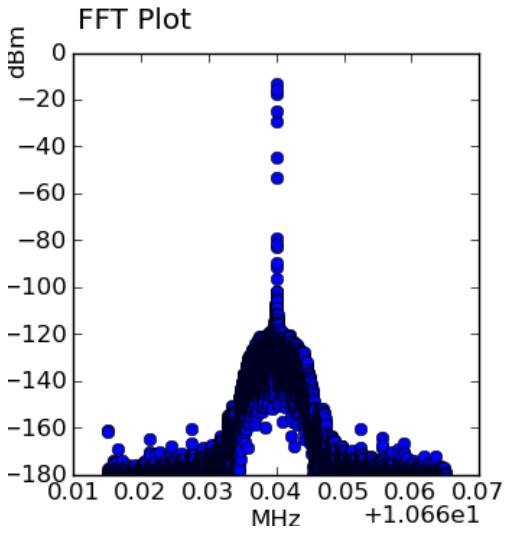


Figura 6.5: GUI - Espectro de frequência do sinal de teste (10.7 MHz; -10 dBm).

A interface do utilizador inclui também os valores instantâneos das estimativas do sinal. Uma amostra é apresentada na Figura 6.6. As estimativas da figura juntamente com o espectro da Figura 6.5 mostram que os algoritmos para a estimação da amplitude do CO, da CNR e da NSD dão os resultados esperados. A estimação de frequência aparenta ter um pequeno desvio. Se for confirmada a linearidade do algoritmo de estimação de frequência este desvio poderá ser eliminado com a introdução de um *offset* contudo recorde-se que a estimativa de frequência é efectuada sobre o sinal do gerador usando o relógio do kit Ettus-Research pelo que diferenças são sempre esperadas entre os dois (não existe um padrão de frequência comum).

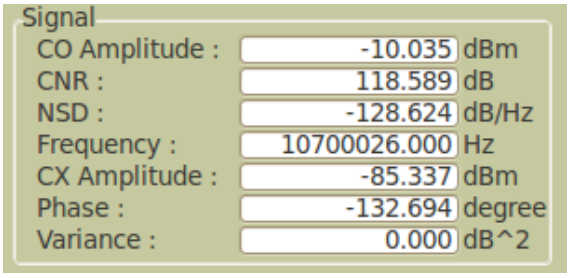


Figura 6.6: Amostra das estimativas do sinal.

Para garantir que a estimação de amplitude do *copolar* está a funcionar correctamente variou-se um pouco a amplitude do sinal de teste para verificar as alterações no seu traçado gráfico (Figura 6.7). A amplitude do sinal de teste sofreu saltos de 10, 20 e 30 dB. O traçado de amplitude do sinal CO acompanhou igualmente estas variações tal como o esperado. De notar que não se encontra nenhum sinal aplicado ao canal do USRP dedicado ao *crosspolar*. As variações observadas no *crosspolar* podem ser devidas a *crosstalk* na *daughterboard* ou mesmo de IF para IF. De qualquer forma faz-se notar que o sinal *crosspolar* é inferior em 70 dB ao *copolar* pelo que não trará consequências na respectiva estimativa de amplitude no cenário experimental esperado. Ainda é possível ver que no pior dos casos o *crosspolar* está sempre abaixo do *copolar* em 40 dB (atenuação máxima de 40 dB que excede largamente a gama dinâmica prevista).

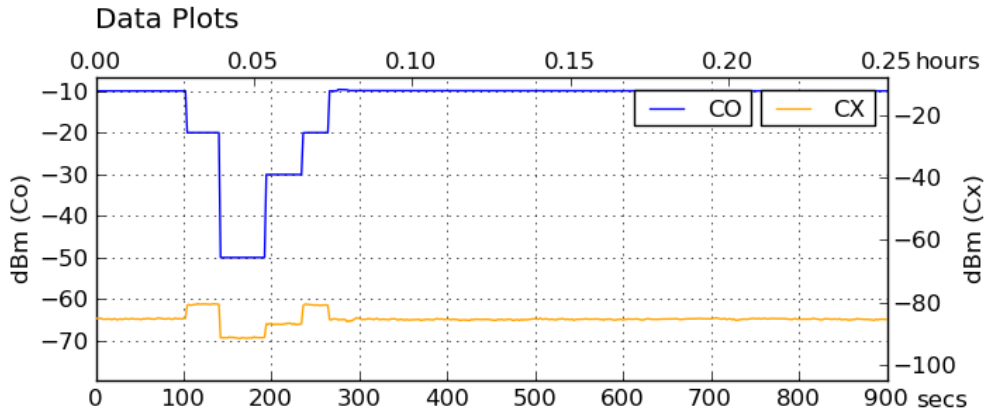


Figura 6.7: Variação de amplitude do sinal de teste.

Um outro ponto que será interessante analisar aqui é o seguimento de frequência. Isto consegue-se com uma experiência semelhante à realizada para a prova do algoritmo de estimação de amplitude do CO, mas neste caso é variada frequência do sinal de teste em vez da amplitude. Como não existe um traçado gráfico com o histórico da frequência, a única forma de mostrar as mudanças desta estimativa é recorrendo aos ficheiros de dados.

Algumas amostras de um ficheiro de dados, relativo à FFT de maior resolução, estão na Figura 6.8. Este ficheiro foi carregado directamente pelo Matlab sem qualquer tipo de modificação ou conversão prévia.

169	170	171	172
-10.0312	-10.0556	-10.0309	-10.0309
-85.7109	-85.8705	-85.8162	-85.8482
-134.4480	-123.5559	-124.0525	-124.6005
118.5154	59.1502	118.9835	118.5502
10700041	10700053	10700051	10700051
6.1610e-08	2.0095e-06	1.9844e-06	1.9599e-06
-128.5466	-69.2059	-129.0144	-128.5811
0	1	0	0
1.3203405410	1.3203e+09	1.3203e+09	1.3203e+09

Figura 6.8: Amostras de um ficheiro de dados.

As amostras registam o momento em que ocorreu um desvio de frequência. Neste caso particular foram somados 10 Hz ao sinal de referência. Este desvio implica a reprogramação do NCO para a nova frequência central e com isto é escrito no ficheiro de dados o valor 1 na penúltima célula da amostra para testemunhar o evento.

Apesar da frequência estimada ter sofrido o mesmo desvio que foi aplicado ao sinal de teste nota-se uma discrepância dos valores da NSD e CNR no momento em que ocorreu a mudança de frequência. Muito provavelmente o facto deve-se ao *hardware* interno do gerador quando concretiza a alteração de frequência. Como com o tempo a frequência do gerador ou do relógio do kit deverão sofrer desvios relativos isto será comprovado registando os dados num funcionamento normal durante algumas horas.

6.2 Teste ao Conjunto Receptor

A partir deste ponto os testes irão ser realizados sobre o conjunto que compõe o receptor de propagação. A primeira abordagem irá ser feita utilizando um sinal de teste "limpo" de um gerador de sinais e na seguinte situação procederemos à introdução de ruído.

6.2.1 Situação 1 – Sinal sem Ruído Adicional

No primeiro conjunto de testes o *hardware* do receptor foi montado na disposição da Figura 6.9.

O sinal de teste, criado pelo gerador de sinais, é dividido em dois sinais com uma diferença de -20 dB entre eles por um acoplador direccional. O sinal de menor potência irá simular o *crosspolar*. Ambos os sinais são depois introduzidos nas cadeias de *hardware* da Figura 6.9 com a particularidade de o sinal que simula o *crosspolar* passar ainda por uma linha de comprimento variável que permitirá alterar a fase entre os dois sinais correlacionados.

Embora se pretenda testar o funcionamento do conjunto na sua globalidade existem alguns pontos, como por exemplo, o isolamento entre cadeias ou a programação do sintetizador que exigem equipamento adicional para teste. Desta forma o analisador de espectros mostrado na figura irá servir como suporte para realizar algumas das medições.

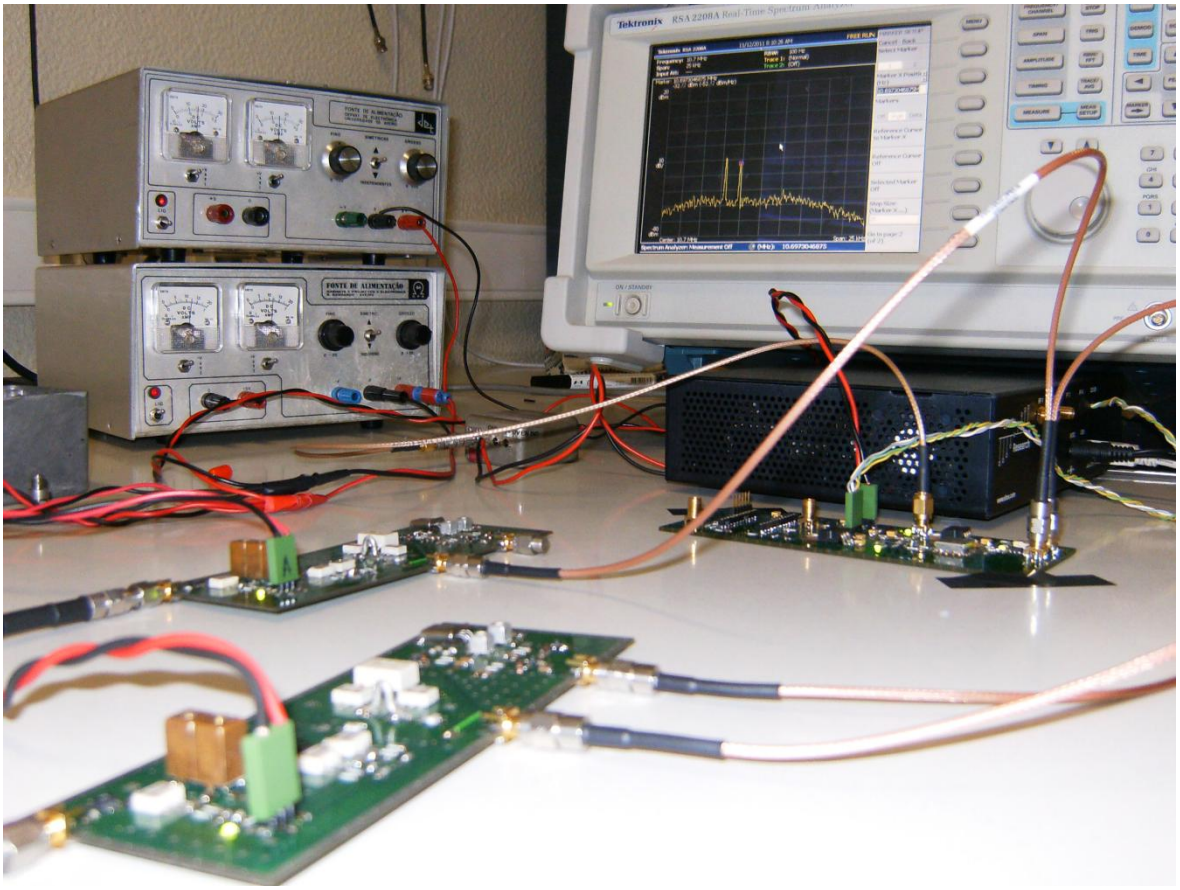


Figura 6.9: Montagem de teste 1 (sem ruído adicional).

6.2.1.1 Isolamento entre as Placas de *Hardware* e Imunidade a Sinais Exteriores

Num sistema RF os sinais que passam de uma placa de *hardware* para a outra são um problema que requer alguma atenção. No nosso caso em particular, iremos ter duas unidades para amplificação e conversão de frequência a funcionar em paralelo que serão respeitantes ao sinal *copolar* e à componente despolarizada *crosspolar*. Será importante analisar o isolamento que existe entre estas duas unidades.

O procedimento para este estudo é bastante simples: é colocado um sinal numa das cadeias de *hardware* terminando a sua saída com uma carga adaptada e é medida a quantidade de sinal que se obtém na saída da segunda cadeia terminando, da mesma forma, a entrada da segunda cadeia. A diferença entre o valor da potência de saída do sinal na primeira cadeia e a obtida por passagem de sinal na segunda à frequência de trabalho (10.7 MHz) ditará o isolamento.

Mediu-se um isolamento acima de 35 dB (por vezes mais de 45 dB) em cada uma das placas de *down convertion*, um valor que varia um pouco com a distância e posição das placas. Conclui-se que será necessário realizar um futuro isolamento entre estas unidades para minimizar a passagem de sinais de uma placa para a outra.

Não serão só os sinais provenientes do *hardware* vizinho que irão influenciar o sistema. Cada uma das placas está sujeita à radiação proveniente do exterior.

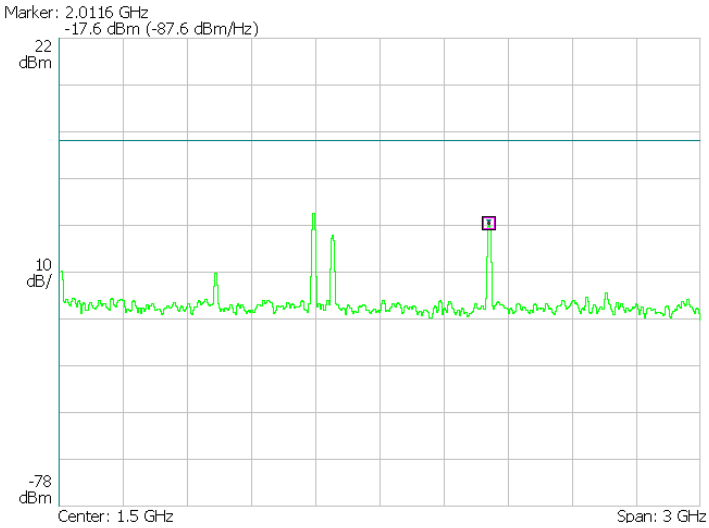


Figura 6.10: *Down converter* 1 - Visualização alargada do espectro.

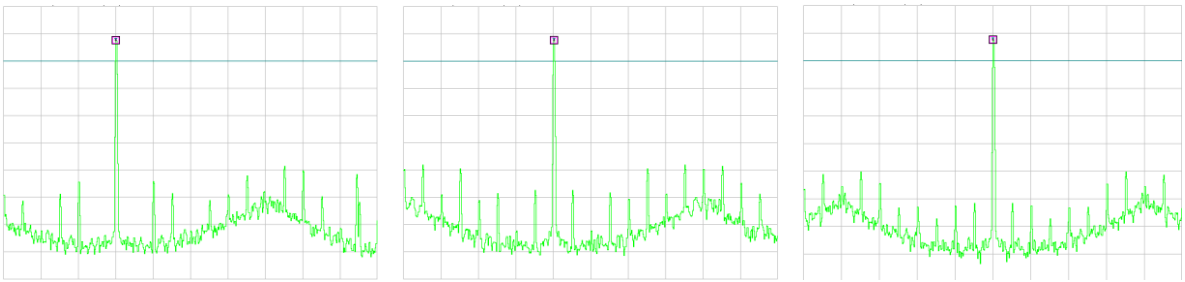
Na Figura 6.10 está o espectro de frequências da saída do *down converter* 1 numa largura de banda de 3 GHz. Além do oscilador local (apontado pelo *marker* da figura), que também irá surgir na saída, existem outros sinais que irão ser introduzidos no *hardware* de forma indevida. Alguns exemplos destes sinais são o GSM (900MHz e 1800MHz), o Wi-Fi (2.4 GHz) ou mesmo as frequências destinadas à distribuição de televisão.

Para evitar que estes sinais influenciem o desempenho do receptor deverá ser realizada uma blindagem do *hardware* através da colocação das placas em caixas apropriadas. Já os 2 GHz do oscilador local poderão ser atenuados com uma melhor filtragem na unidade de 10.7 MHz contudo não inspiram cuidados.

6.2.1.2 Desempenho do Oscilador Local

O oscilador local é um dos elementos mais importantes do receptor. Utilizando a síntese de frequências será possível derivar uma determinada frequência de um sinal de referência.

O LO do receptor de propagação inclui um sintetizador que pode ser programado através da escrita dos seus registos. Isto permite um ajuste de frequência do LO quando for necessário.



a) f = 2.010695 GHz; b) f = 2.010700 GHz; c) f = 2.010750 GHz;

Figura 6.11: Oscilador Local - Várias frequências.

Utilizado o módulo *reg_calc.py* do *software* do detector realizou-se a programação do sintetizador para várias frequências. Alguns dos sinais obtidos à saída da unidade de síntese estão na Figura 6.11. Em termos de frequência e potência do sinal os resultados conseguidos encontram-se dentro do esperado.

Um outro aspecto importante que aqui se deve ter em conta é a presença de espúrias à volta do sinal. Estas espúrias são resultado do processo de síntese e a sua posição e potência varia consoante a frequência sintetizada (Figura 6.11). A existência destas espúrias poderá levar à presença de sinais indesejados próximos da IF_2 que poderão confundir o detector e resultar em estimativas erradas em algumas condições muito particulares.

Na Figura 6.12 pode inferir-se a resposta em frequência do filtro a cristal da IF_2 pela formatação aplicada ao ruído térmico O sinal visto no espectro da figura mostra o problema referido no último parágrafo. Esta risca não é o sinal RF de teste mas sim uma espúria resultante do processo de síntese. O sinal espúrio pode causar problemas caso a frequência esteja muito próxima do sinal (pode a sua potência ser detectada como sinal) ou fique dentro da gama usada para estimar a NSD. Portanto recomenda-se uma avaliação criteriosa dos parâmetros do sintetizador na aplicação final. A questão em causa é a relação da potência da espúria relativamente ao sinal e não apenas em relação à portadora do LO. Este problema das espúrias não é apenas do sintetizador: verificámos que os próprios geradores as produziam.

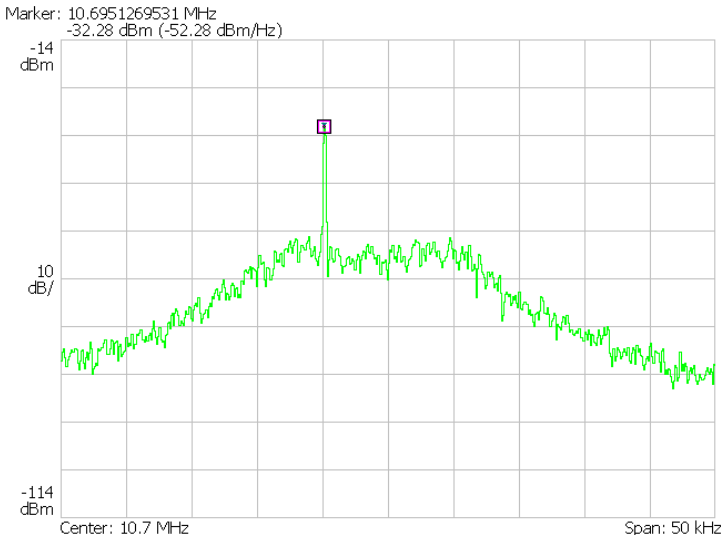


Figura 6.12: Resposta do filtro a cristal do *down converter* 1.

O problema das espúrias poderá ser solucionado, ou pelo menos minimizado, optimizando os parâmetros de programação do sintetizador.

6.2.1.3 Funcionamento Global

O próximo passo será o teste global do receptor com sinais ainda sem ruído aditivo. O USRP terá à sua entrada dois sinais que representarão o *copolar* e o *crosspolar* que provêm do *frontend* analógico do receptor. O acompanhamento das variações dos sinais

será feito com o *software* do detector que tratará não só de mostrar as alterações em tempo real na GUI como armazenará toda a informação em ficheiros binários.

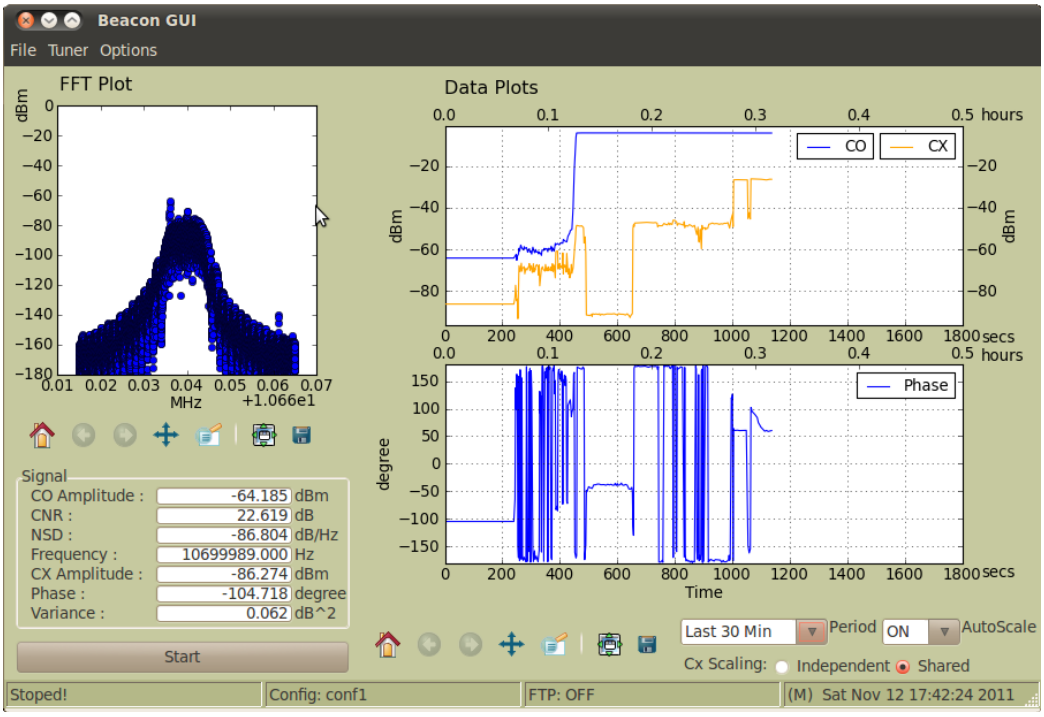


Figura 6.13: Interface do utilizador em funcionamento - situação 1.

Na Figura 6.13 encontra-se a GUI com traçados gráficos cujas variações correspondem a vários testes realizados ao longo do tempo. Para uma melhor descrição dos eventos vamos centrar-nos nos traçados gráficos de amplitude e fase (Figura 6.14).

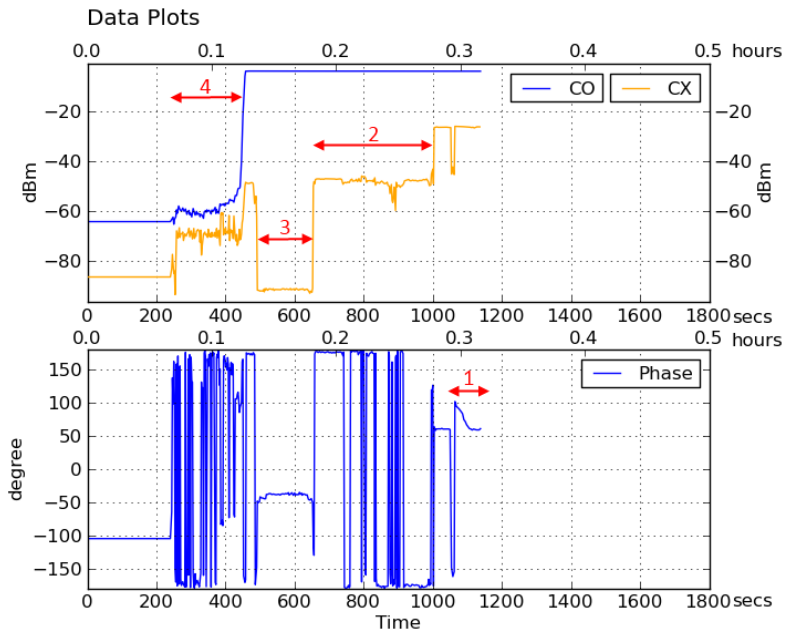


Figura 6.14: Traçados de amplitude e fase dos sinais CO e CX - Várias situações.

Na situação 1, Figura 6.14, a variação de fase apresentada é devida à modificação do comprimento de uma linha de comprimento variável com uma excursão de cerca de 1.6 cm a qual corresponde um desfasamento de cerca de 40^{o} que é confirmado aproximadamente pela mesma figura.

O caso 2, onde ocorre uma descida da amplitude do sinal *crosspolar* sem modificação do *copolar*, mostra o fenómeno de *crosstalk* que ocorre entre as cadeias do CO e do CX. Nesta situação foi retirado o sinal da entrada da cadeia do *crosspolar*. O sinal CX que surge é, em grande parte, devido à passagem do sinal da cadeia do *copolar* para a cadeia do *crosspolar*. A variação de fase entre -180^{o} e $+180^{o}$ deve-se ao facto da amplitude do *crosspolar* ser bastante reduzida e o vector rodear a origem das coordenadas.

Além do *crosstalk* entre as cadeias do CO e do CX, no próprio URSP também existe passagem de sinal entre canais. No ponto 3 da figura o sinal correspondente ao *crosspolar* foi retirado do seu canal do USRP e esta entrada foi terminada com uma carga adaptada. Como se pode verificar pela amplitude do *crosspolar* o isolamento entre canais no USRP é bastante grande (> 90 dB) e não será um problema significativo.

A última situação, o ponto 4, mostra as estimativas de amplitude do sinal *copolar* e sinal *crosspolar* com a diminuição da amplitude do sinal de teste. As variações de amplitude são correctamente seguidas pelo sinal *copolar* numa gama superior a 30 dB. No caso do sinal *crosspolar* quando o sinal baixa demasiado de amplitude a sua estimativa começa a ter variações muito significativas por se encontrar muito próximo do patamar de ruído.

Com um nível de sinal baixo à entrada do receptor verificou-se ainda se o detector estaria de facto a "agarrar-se" ao sinal RF ou uma espúria. Voltando à Figura 6.13, olhemos para a janela com o espectro de frequência do sinal *copolar*. Na banda de passagem do filtro FIR encontram-se dois sinais com amplitudes diferentes. O sinal de maior amplitude é a espúria já conhecida do subcapítulo anterior. Como se pode verificar o espectro encontra-se centrado na risca de menor amplitude (a risca que representa o sinal) o que mostra que os mecanismos implementados no *software* em casos de baixa CNR se encontram a funcionar como esperado.

Um outro aspecto importante é o nível de sinal de entrada aceite pelo USRP. Com um sinal de 10 dBm na entrada provou-se existir saturação no *hardware* do kit. Para evitar erros nas medidas devido a saturação será aconselhável introduzir sinais com um nível de amplitude na ordem dos 0 dBm nos canais do USRP.

6.2.1.4 Linearidade

Vamos agora avaliar a linearidade do sistema. A principal intenção do receptor será acompanhar as variações de amplitude do sinal do *beacon* do Alphasat ao longo do tempo, por tal iremos proceder à descida de amplitude do sinal de teste e verificar o acompanhamento do detector em termos da amplitude do *copolar* e do *crosspolar*.

O sinal de teste irá partir do gerador de sinais com uma amplitude de -83 dBm de forma a evitar saturação do USRP. O sinal *crosspolar* estará 20 dB abaixo do *copolar*.

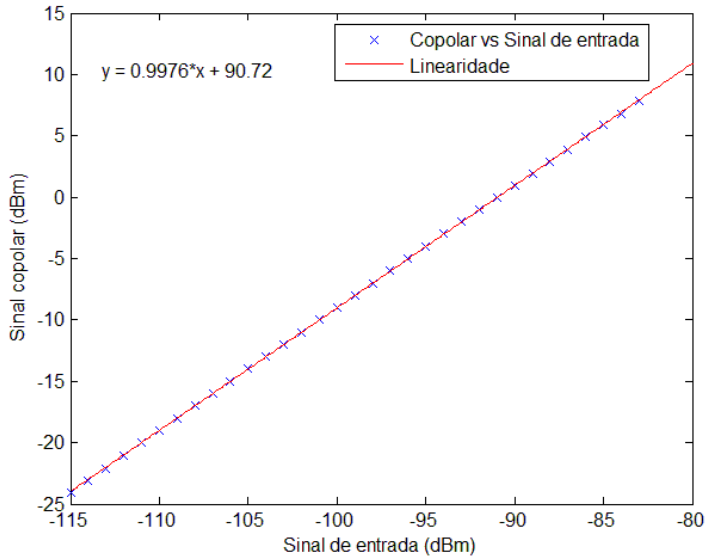


Figura 6.15: Variação de amplitude do sinal *copolar* - linearidade.

Em relação ao sinal *copolar* (Figura 6.15) os resultados obtidos encontram-se bastante próximos do ideal. A curva de linearidade encontram-se muito próxima dos pontos recolhidos na gama em análise (32 dB) o que mostra a alta linearidade que o sistema apresenta para esta estimativa.

Quanto ao sinal *crosspolar*, uma vez que este tem uma potência mais baixa que o *copolar*, a gama de valores para os quais o sistema foi avaliado está representada na Figura 6.16 que corresponde a uma atenuação máxima do *copolar* de 26 dB. Contudo da interface gráfica (não apresentada) é visível que a amplitude relativa do *crosspolar* acompanha o *copolar* até mais de 35 dB.

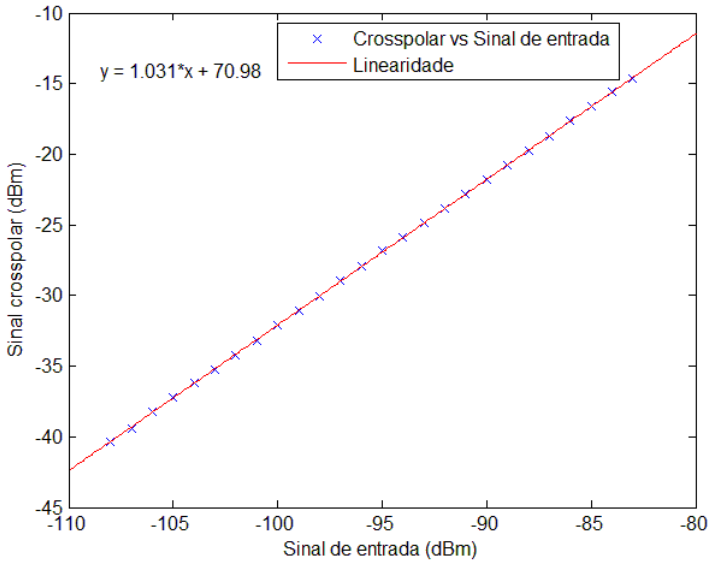


Figura 6.16: Variação de amplitude do sinal *crosspolar* - linearidade.

Para esta gama de valores a estimativa de amplitude do *crosspolar* é também bastante linear e assim tanto o algoritmo de cálculo de amplitude do *crosspolar* como o do *copolar* apresentam os resultados desejados.

6.2.2 Situação 2 – Sinal com Ruído Adicionado

Chegámos agora à situação em que o receptor de propagação será posto à prova com um sinal de teste realista, ou seja, com a introdução de um patamar de ruído adicional que simulará o ruído do sinal do *beacon*.

A montagem feita para este último conjunto de testes encontra-se na Figura 6.17.

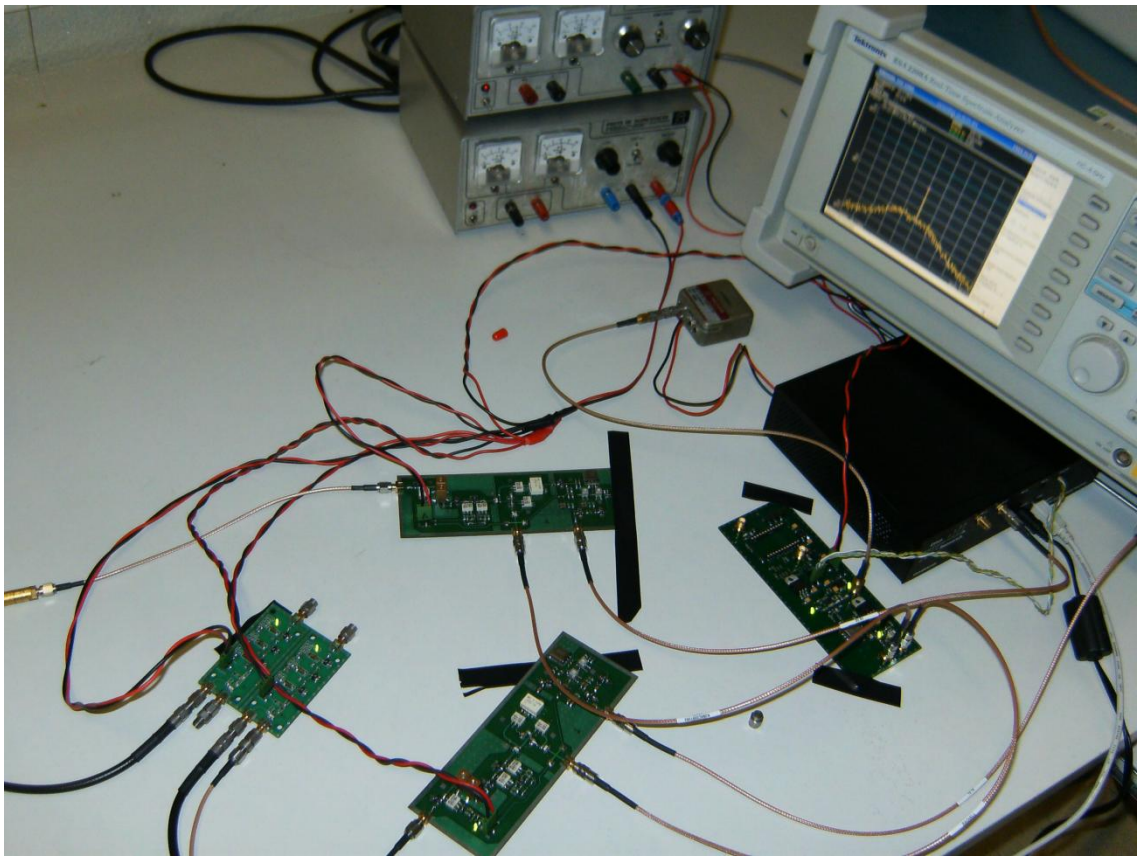


Figura 6.17: Montagem de teste 2 (com ruído adicional).

Embora tenha sido realizado um dimensionamento adequado do gerador de ruído, verificou-se um imprevisto: a adição do ruído causa uma saturação das unidades de acondicionamento e conversão do sinal observando-se uma perda de ganho da cadeia. O gerador de ruído gera de facto ruído com uma distribuição espectral sensivelmente constante até mais de 8 GHz e portanto potência em demasia. Para resolver esta questão seria desejável incluir à saída do gerador de ruído um filtro passa-banda. O sinal recebido de um qualquer *frontend* de um sistema de telecomunicações tem aliás sempre um carácter passa-banda. Um filtro com uma largura de 150 MHz reduziria a potência de ruído em pelo menos 17 dB.

Ainda assim procedeu-se à adição de ruído ao sinal de teste e colocaram-se atenuadores à saída do gerador de ruído. O sinal *copolar* inicial que entrará no USRP terá uma CNR de cerca de 52 dB e o *crosspolar* uma CNR que ronda os 30 dB.

Apesar de não conseguirmos o sinal com o nível de amplitude pretendido a CNR é a adequada para avaliar o desempenho do receptor contudo a amplitude do sinal à entrada do USRP é bem menor que o pretendido.

6.2.2.1 Funcionamento Global

Em termos de funcionamento, o *software* do detector manteve um comportamento similar ao da primeira situação com o sinal de teste "limpo" (Figura 6.18).

O sinal *copolar* seguiu correctamente as variações de amplitude do sinal de entrada e a detecção do sinal continuou a ser realizada como era pretendido.

Relativamente ao sinal *crosspolar* é visível uma discrepância no traçado da Figura 6.18 que poderá indicar a presença de um erro no algoritmo da estimativa de amplitude do sinal *crosspolar* que aliás funciona sem problemas nos testes realizados sem ruído. No entanto, fora desta gama irregular e apesar da baixa potência, o sinal *crosspolar* acompanha as variações de amplitude do sinal de entrada (repare-se que as escalas do *copolar* e *crosspolar* são independentes).

A variação da fase relativa entre os dois sinais apresentada no gráfico deve-se à baixa potência do sinal *crosspolar*.

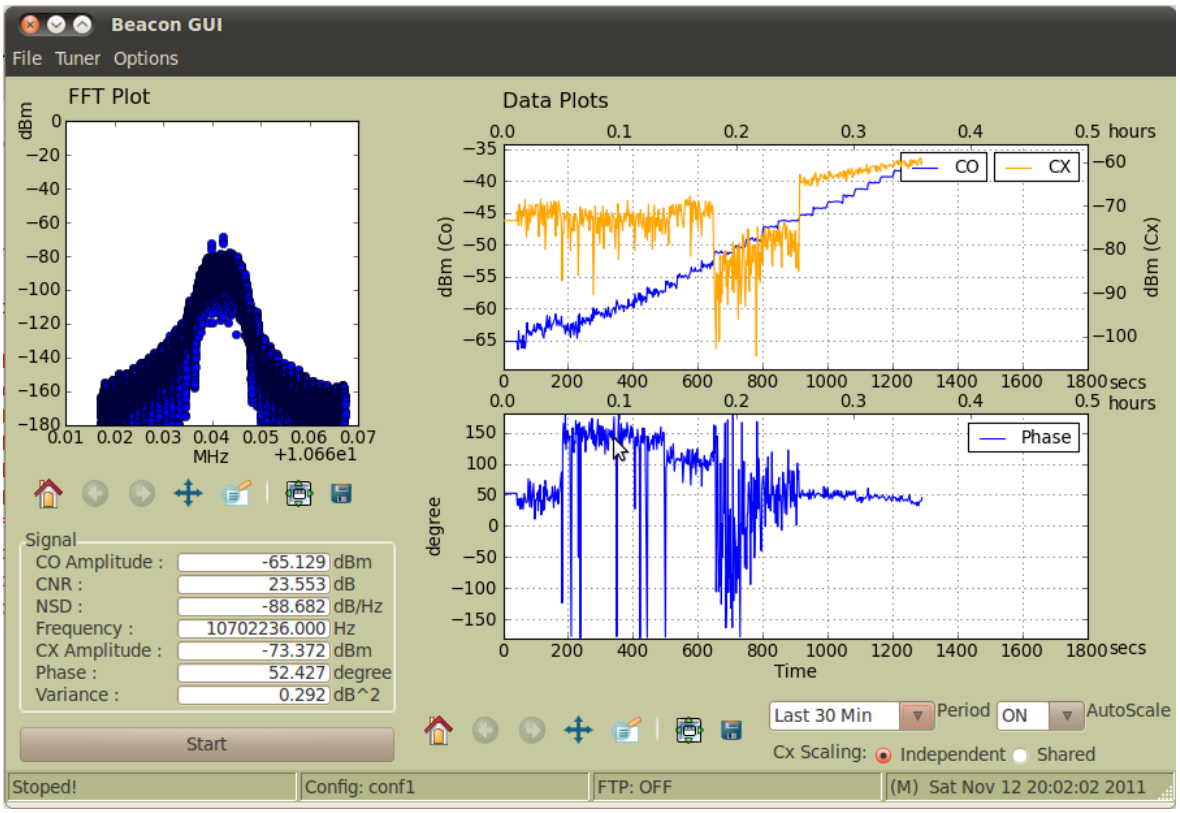


Figura 6.18: Interface do utilizador em funcionamento - situação 2.

6.2.2.2 Linearidade

Com as novas condições do sinal de teste o detector digital terá agora que lidar com um caso não ideal e que se aproxima de uma situação realista. Desta forma será feita mais uma vez a análise da sua linearidade.

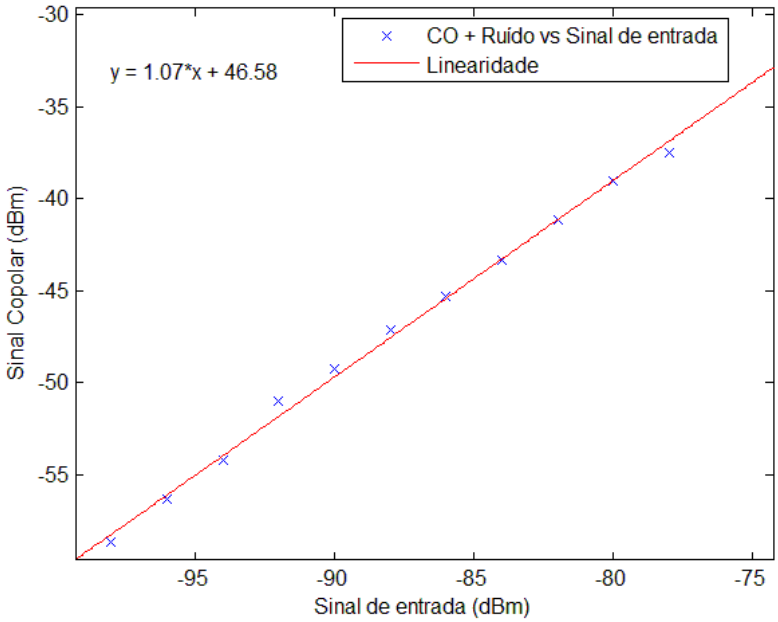


Figura 6.19: Variação de amplitude do sinal *copolar* com ruído adicional - linearidade.

De acordo com o gráfico da Figura 6.19 podemos verificar que, apesar de não ser tão elevada como na primeira situação (6.2.1.4), a linearidade em termos da resposta de amplitude do sinal *copolar* continua a ser razoavelmente assegurada. Apesar das condições mais extremas o sistema mostra-se capaz de acompanhar as variações do sinal de teste cumprindo os requisitos que foram impostos.

Estes testes foram realizados com algumas limitações experimentais devido à variação de amplitude do gerador se fazer pelos botões de controlo. Isto é, não está garantida uma variação contínua de amplitude (o gerador deve comutar internamente atenuadores de uma forma que não conhecemos) que será a esperada nas medidas de propagação.

7 Conclusões Finais e Trabalho Futuro

7.1 Conclusões

Podemos concluir que os objectivos propostos neste trabalho foram, de forma geral, cumpridos.

As novas adições ao *frontend* analógico do receptor mantêm o *hardware* a funcionar como pretendido, cumprindo com as novas características do sinal. Embora se destaque o bom funcionamento verificou-se que para um desempenho óptimo será importante proceder à colocação das unidades do *frontend* em caixas adequadas para reduzir interferências externas e passagem de sinais entre as cadeias de *hardware*.

O gerador de ruído construído para os testes, à falta de geradores de ruído, provou ser muito útil contudo é necessário limitar a largura de banda de saída a qual pode ser feita com filtros centrados à frequência de teste.

A nível do *software* do detector as correcções realizadas bem como as novas implementações tornaram o programa mais estável e mais completo. A nova abordagem utilizada na interface gráfica mostrou ser uma boa solução pela qualidade dos traçados e interacção com utilizador. Os vários parâmetros do sistema são agora configuráveis de forma simples e directa através dos menus da GUI e os mecanismos de aviso do utilizador em caso de situações críticas funcionam como pretendido mantendo a estabilidade do detector.

A revisão dos algoritmos de cálculo resultou em estimativas mais precisas tanto das amplitudes como da frequência e igualmente da NSD. Ficou também provado que os algoritmos implementados para situações de baixa CNR estão a funcionar.

Os ficheiros de dados binários podem agora ser carregados directamente no Matlab sem necessidade de conversão e a sua escrita durante a aquisição é realizada com o cuidado de evitar corrupção de dados em caso de interrupção abrupta. Isto facilita bastante o carregamento dos dados para análise *off-line* e mesmo para inspecções fortuitas da qualidade dos dados adquiridos que é fundamental para detectar anomalias que por vezes não são evidentes.

Além do já referido é importante salientar os conhecimentos adquiridos em vários tópicos bastante transversais. Sejam as comunicações por satélite, a síntese de frequências, a electrónica de RF, o processamento digital de sinal ou mesmo a utilização do conjunto GNU Radio/USRP e linguagens de programação C++, Python e wxPython as várias matérias envolvidas no projecto resultam num conjunto de competências que serão importantes para o futuro.

Salienta-se que no âmbito do projecto COST-IC0802 vários têm sido os investigadores que nos têm contactado sobre o actual desenvolvimento que estamos a realizar. Há alguns grupos que tencionam participar na experiência de propagação Alphasat e estão a adquirir kits Ettus Research (por exemplo o Josef Stefan Institute, Eslovénia e uma empresa na Noruega). A Universidade de Vigo também desenvolveu um receptor baseado em detecção espectral usando o kit SDR-IQ [52] que contudo não achámos

apropriado pois apenas oferece 1 canal e não permite a programação em tempo real do NCO. Uma colaboração que se perspectiva é com a UPM-ETSI que igualmente pretende avançar com uma solução SDR para a experiência Alphasat.

7.2 Trabalho Futuro

Como trabalho futuro será importante:

- Colocar as placas do *frontend* em caixas blindadas para evitar interferências externas e minimizar o *crosstalk* entre as cadeias de *hardware*;
- Colocar o gerador de ruído numa caixa blindada para evitar interferências externas e desenvolver filtros passa-banda para colocar na saída conforme a aplicação pretendida;
- Realizar testes para verificar a estabilidade do sistema de recepção com a variação da temperatura;
- Optimizar o mixer de rejeição de imagem;
- Reconfigurar os parâmetros da programação do sintetizador para minimizar as espúrias do processo de síntese de frequências;
- Permitir a configuração do sintetizador para uma resolução de frequência arbitrária usando um oscilador de referência com frequência igualmente arbitrária;
- Corrigir pequenas falhas no programa do detector;
- Implementar um mecanismo para controlo de motores de apontamento da antena através das interfaces do USRP.

Bibliografia

[1] ELBERT, BRUCE R. - Introduction to satellite communication: The Artech House space applications series. 3rd. Boston: Artech House, 2008.

[2] How satellites work. http://www.cnes.fr/web/CNES-en/1870-how-satellites-work.php. Acedido em Setembro de 2011.

[3] Types and uses of satellites. http://www.satellites.spacesim.org/english/engineer/copy/index.html. Acedido em Setembro de 2011.

[4] A communication satellite payload. http://www.aero.org/publications/crosslink/winter2002/01_sidebar3.html. Acedido em Setembro de 2011.

[5] ViaSat. http://www.viasat.com/broadband-satellite-networks/viasat-1. Acedido em Outubro de 2011.

[6] Eutelsat. http://www.eutelsat.com/satellites/9e_ka-sat.html. Acedido em Outubro de 2011.

[7] Yahsat. http://www.yahsat.ae/Kabroadband.htm. Acedido em Outubro de 2011.

[8] SELDING, PETER B. DE- Telesat Sees Slight Increase in Quarterly Sales and Profit. http://www.spacenews.com/satellite_telecom/110506-telesat-higher-sales-profit.html, 2011.

[9] Alphasat. http://telecom.esa.int/telecom/www/object/index.cfm?fobjectid=1138. Acedido em Setembro de 2011.

[10] Inmarsat. http://www.inmarsat.com/About/?language=EN&textonly=False. Acedido em Setembro de 2011.

[11] Alphasat TDP5. http://telecom.esa.int/telecom/www/object/index.cfm?fobjectid=26445. Acedido em Setembro de 2011.

[12] REED, JEFFREY HUGH - Software radio : A modern approach to radio engineering: Prentice Hall communications engineering and emerging technologies series. Upper Saddle River, NJ: Prentice Hall, 2002.

[13] Altera SDR. http://www.altera.com/end-markets/wireless/advanced-dsp/sdr/wir-sdr.html. Acedido em Setembro de 2011.

[14] SOUSA, RICARDO; PIRES, ANDRÉ - Receptor Digital para Medição de Balizas de Satélite. Aveiro: University of Aveiro, 2006.

[15] RODRIGUES, EMANUEL FILIPE DE ORNELAS- Software GNU Radio para Detector Digital de Sinais CW com baixa SNR. Tese de Mestrado em Eng. Eléctrónica e Telecomunicações, 2009.

[16] CABRAL, EMANUEL MOURA- Unidade Interior de Receptor de Satélites Baseado em kit Rádio Digital. Tese de Mestrado em Eng. Eléctrónica e Telecomunicações, 2009.

[17] Mini-Circuits. http://www.minicircuits.com/homepage/homepage.html. Acedido em Setembro de 2011.

[18] Mixer Image Frequency. http://www.rfcafe.com/references/electrical/image-frequency.htm. Acedido em Outubro de 2011.

[19] ROGERS, JOHN; PLETT, CALVIN - Radio Frequency Integrated Circuit Design: Artech House microwave library. Boston: Artech House, 2003.

[20] LEESON, D.B. - A Simplified Model of Feedback Oscillator Noise Spectrum. Proceedings of the IEEE, 1965.

[21] MINI-CIRCUITS - VCO Designer's Handbook. 1996.

[22] KROUPA, V. F. - Phase Lock Loops and Frequency Synthesis. John Wiley & Sons Ltd., 2003.

[23] MOON, SUNG TAE; VALERO-LÓPEZ, ARI YAKOV; SÁNCHEZ-SINENCIO, EDGAR - Fully Integrated Frequency Synthesizers: A Tutorial. International Journal of High Speed Electronics and Systems. World Scientific Publishing Company.

[24] Analog Devices. http://www.analog.com. Acedido em Outubro de 2011.

[25] KUNDERT, KEN- Predicting the Phase Noise and Jitter of PLL-Based Frequency Synthesizers. Designer's Guide Consulting, Inc., 2009.

[26] LUDWIG, REINHOLD; BRETCHKO, PAVEL - RF Circuit Design - Theory and Applications. Prentice Hall, 2000.

[27] Miteq. http://www.miteq.com. Acedido em Setembro de 2011.

[28] Ettus Research LLC. http://www.ettus.com. Acedido em Setembro de 2011.

[29] HAMZA, FIRAS ABBAS- The USRP under 1.5X magnifying lens!, 2008.

[30] SMITH, STEVEN W. - The Scientist and Engineer's Guide to Digital Signal Processing. 1998.

[31] DEKKER, A.P.- The Exchange of Bits Against Bandwith.

[32] Enhancing ADC resolution by oversampling. Atmel Application Note, 2005

[33] BRANNON, BRAD; CLONINGER, CHRIS- Redefining the Role of ADCs in Wireless. Analog Devices, Inc., 2001.

[34] LYONS, RICHARD G. - Understanding digital signal processing. Prentice Hall, 2001.

[35] Matlab. http://www.mathworks.com/products/matlab/index.html. Acedido em Setembro de 2011.

[36] The Fundamentals of FFT-Based Signal Analysis and Measurement in LabVIEW and LabWindows/CVI. National Instruments, 2009.

[37] GNU Radio. http://gnuradio.org. Acedido em Outubro de 2011.

[38] PILGRIM, MARK - Dive Into Python. 2004.

[39] SWIG. http://www.swig.org. Acedido em Outubro de 2011.

[40] wxPython. http://www.wxpython.org. Acedido em Outubro de 2011.

[41] Tutoriais Python e wxPython. http://zetcode.com. Acedido em Outubro de 2011.

[42] RAPPIN, NOEL; DUNN, ROBIN - wxPython in Action. Manning Publications Co., 2006.

[43] PRECORD, CODY - wxPython 2.8 Application Development Cookbook. Packt Publishing, 2010.

[44] Matplotlib. http://matplotlib.sourceforge.net. Acedido em Outubro de 2011.

[45] TOSI, SANDRO - Matplotlib for Python Developers. Packt Publishing, 2009.

[46] ZVEREV, ANATOL I. - Handbook of filter synthesis. New York, : Wiley, 1967.

[47] Neosid. http://www.neosid.de. Acedido em Outubro de 2011.

[48] KRAUSS, HERBERT L.; BOSTIAN, CHARLES W.; RAAB, FREDERICK H. - Solid state radio engineering. New York: Wiley, 1980.

[49] SMITH, JACK - Modern communication circuits: McGraw-Hill series in electrical and computer engineering. Boston, Mass: WCB/McGraw-Hill, 1998.

[50] Dropbox. http://www.dropbox.com. Acedido em Setembro de 2011.

[51] MAT-File Format. The MathWorks, Inc., 2011.

[52] SDR-IQ Receiver. http://www.rfspace.com/RFSPACE/SDR-IQ.html. Acedido em Novembro de 2011